Local Geometrically Enriched Mixtures for Stable and Robust Human Tracking in Detecting Falls

Regular Paper

Michalis Kokkinos¹, Nikolaos D. Doulamis¹ ² and Anastasios D. Doulamis²

¹ National Technical University of Athens, Zografou, Athens, Greece
² Technical University of Crete, Kounoupidania, Chania Greece

* Corresponding author E-mail: ndoulam@cs.ntua.gr

Received 20 Jun 2012; Accepted 3 Oct 2012

DOI: 10.5772/54049

© 2013 Kokkinos et al.; licensee InTech. This is an open access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/3.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Abstract Detecting a fall through visual cues is emerging as a hot research agenda for improving the independence of the elderly. However, the traditional motion-based algorithms are very sensitive to noise, reducing fall detection accuracy. Another approach is to efficiently localize and then track the foreground object followed by measurements that aid the identification of a fall. However, to perform robust and stable tracking over a long time is a challenging research aspect in computer vision society. In this paper, we introduce a stable human tracker able to efficiently cope with the trade-off between model stability (accurate tracking performance) and adaptability (model evolution to visual changes). In particular, we introduce local geometrically enriched mixture models for background modelling. Then, we incorporate iterative motion information methods, constrained by shape and time properties, to estimate high confidence image regions for background model updating. This way, we are able to detect and track the foreground objects even when visual conditions are dynamically changed over time (luminosity or background/foreground changes or active cameras).

Keywords Human Tracking, Local Geometry, Visual Fall Detection, Background Modelling, Constrained Shape-Time Motion Analysis

1. Introduction

Traumas resulting from falls amongst elderly people have been reported as the second most common cause of death and the third most common cause of chronic disability, based on the records of the World Health Organization [1]. The proportion of people sustaining at least one fall in one year varies from 28% to 35% for ages of 65 and over [2], while falls often signal the “beginning of the end” of an older person’s life. Additionally, fall traumas often cause major movement impairments with concomitant consequences for their lives and surroundings.

Thus, as falls and fall-related injuries remain a major challenge in the public health domain, reliable and immediate detection of falls is important so that adequate medical support can be delivered. Today, the existing algorithms use either personal embedded sensors or
video sensors [3]. Embedded sensors present the drawback that they should be worn by the individuals, which is not convenient, especially for persons with mild dementia. Instead, cameras offer transparent monitoring and also provide additional information to the caregivers, such as evaluation of the type of the fall. However, detecting falls is a salient research issue in computer vision society due to the complexity of the problem as far as the visual content is concerned. The algorithm should ideally (a) detect falls in real time (or at least just in time), i.e., without losing the resolution accuracy for the fall detection, (b) be tolerant against other daily activities, (c) be robust to background and illumination changes, (c) give accurate results when more than one people are present in the scene, (d) identify falls regardless of the position in relation to the camera and (e) be tolerant to camera changes (active cameras).

1.1 Previous Works

Personal embedded sensors have been extensively studied in the context of fall detection. Examples include RFID architectures, goniometers, gyroscopes and especially accelerometers [4]-[6]. The developed systems are of small size and of low consumption, making them affordable and easy to wear for the elderly (see, for example, the FALLWATCH project: http://www.fallwatch-project.eu/). Other types of algorithms use infrared [7] switches in users’ shoes [8], vibration and other sound effects caused by the collapse of the body on the floor [9], or even a combination of these [10].

Visual inspection of a fall through the use of cameras is another alternative. Algorithms for visual fall detection present a series of advances compared to the current approaches that utilize specialized devices. Firstly, fall alerts are activated in a transparent and seamless way for the elderly - there is no need to carry out or wear specialized devices/sensors. Secondly, the method is suitable for people who suffer from cognitive/mental diseases like mild dementia, meaning they are unwilling or forget to wear the specialized devices all the time. Privacy is not spoiled since these tools allow the selective display and transmission of visual information which presents only fall events while being prevented from delivering visual media content.

Generally, two different approaches have been proposed in the literature for visual fall detection. The first focuses on estimating human motion in complex background conditions, followed by machine learning methodologies which distinguish falls from other daily human activities (sitting, bending, moving, standing, etc). The second concentrates on localizing the foreground object(s) even from complex and dynamic background content. The main problem in the first case is that motion information is usually insufficient for distinguishing a fall from other human actions. A person may fall in all possible directions in relation to the camera (e.g., towards the camera, in the opposite direction, across the camera, oblique fall, etc.). On the other hand, foreground extraction provides a flexible procedure for fall detection, although such a task in highly dynamic visual environments is a challenging aspect.

A characteristic work for fall detection is the Asynchronous Temporal Contrast (ATC) vision sensor [11]. The ATC sensor calculates centroids as motion information and falls are alerted when significant vertical velocity is detected. The work of [3] proposes a multiview camera system for detecting falls combined with a Layer Hidden Markov Model (LHMM). A motion-based fall detection algorithm that operates on the video compressed domain has been presented in [12]. However, these approaches, apart from being so sensitive to motion noise, also fail to detect falls when they occur in some directions (e.g., oblique). A pattern analysis algorithm that discriminates fall events from slip events using energy maps is reported in [13]. 3D active vision has been proposed in [14] and in [15]. Finally, combination of visual sensors with kinematic sensors so as to improve the system performance has been proposed in [16] and in [17].

[18] discusses an alternative methodology in which the person’s body is modelled as ellipses, and falls are detected by exploiting human shape variations via support vector (SVM) classifiers. Video analytics from shape and motion features have been reported in [19]. In the same framework, the work of [20] introduces a human shape deformation approach, based on the use of Procrustes distance combined with classification methods to distinguish a fall from normal human activities. Classification methods for fall versus activity have been reported in [21]. However, such approaches exploit a multiple camera configuration scheme to address occlusions since they do not incorporate object tracking methodologies.

An alternative approach is to incorporate object tracking tools towards an efficient detection of a fall [22]. However, the main difficulty is that object tracking should be robust to significant environmental visual changes and stable for long periods; otherwise the adopted system cannot be applied for a commercial use. In [23], a background subtraction approach is adopted and the falls are distinguished from other human activities through cascading multi-SVM classifiers. This approach, however, still considers almost static background.

Assuming continuous monitoring (24 hours per day, seven days per week), we need a tracking re-initialization technique to automatically “re-configure” the tracker to
avoid error accumulation [24]. A dynamic and self-adaptive background modelling algorithm for detecting falls is discussed in [25]. This algorithm is based on a single-pixel modelling approach and thus it is likely to yield confused results, especially when similar colour patterns in the background also appear in the foreground object. For this region, in this paper, we propose a local geometrically enriched self-adaptive background modelling in order to provide a stable and robust foreground tracking for long periods even despite significant visual environmental changes.

1.2 Our Contribution/Innovation

In this paper, we propose a new fall detection scheme that exploits visual observations. The main advantages of the proposed scheme are that we adopt unconstrained and simple requirements.

1.2.1 Visual Constraints

First, we use single visual cameras of low cost and low resolution. Such selection makes the system applicable on a large scale. Second, visual conditions are dynamically changed. Objects and furniture can change position in the scene, humans can perform any regular activity, (sitting, moving objects, bending, sleeping, standing, etc.), illumination conditions fluctuate (lights can be on or off, sun‐light reflections from windows can change pixels’ luminosity values, lights can be reflected/refracted in glasses (mirrors), humans can be moved in all directions in the room, etc.). In other words, we implement our approach in real-world visual conditions where all the daily human activities can take place and no particular constraints are imposed on visual information. Finally, the developed algorithm operates in real time, or at least just in time. The computer vision tools have been developed using Intel's Integrated Performance Primitives tools, exploiting processor hardware capabilities.

We also impose no constraint on the direction of the falls. Again, the problem is very challenging since we assume that the persons can move in any position within the scene, and thus falls can occur not only in vertical directions (towards the camera) but also in oblique directions (moving across the camera). It is also difficult to deal with a fall in a direction opposite to the camera view, since in this case, due to perspective projection, the fall can be easily confused with other human actions such as sitting or lying.

The proposed scheme is evaluated in real-life conditions in different places and over long periods, in the framework of European Union funded project [25].

1.2.2 Technical Innovations

All these aspects constitute a challenging visual environment. To derive efficient fall detection in such an environment, we combine on the one hand adaptive background models able to capture slight modifications of the background patterns with, on the other hand, motion-based algorithms that define with high accuracy which parts of an image should be considered as foreground/background when abrupt, non-period and sudden changes take place. In this paper, a background subtraction methodology is adopted to tackle the problem of tracking over very long periods. In contrast to previous, conventional works such as [25], we adopt probabilistic mixture models like the Gaussian Mixture Models (GMMs), which exploit geometric properties of locally connected regions. Our approach adopts a graph-based saliency map methodology so that the most important pixels are selected (pixels in which humans feature more than others), and these to be fed as GMM inputs.

This constitutes a major innovation of the proposed approach compared to other background modelling techniques, especially in cases of very dynamic background content. Most conventional background subtraction methods assume movement in the background. But this movement is rather simple, often oscillated and mostly low compared to the foreground movement. In this paper, due to the continuous operation of the camera, such assumptions are not valid. Background content can change dramatically from time to time (we cannot impose a static background in someone’s home). The inclusion of local connectivity in modelling the background content increases robustness and tolerance to noise owing either to camera defects and/or illumination fluctuation.

Having detected the moving object, we then model the foreground as a moving object. To this end, we combine optical flows techniques with the good features to track methodology of [26]. Then we define the time instances required for updating the background as well as the frames regions in which such updating is required. Shape and motion constraints are imposed to improve the analysis.

The results have been evaluated in real-world conditions and over a long time, revealing the robustness, reliability and high performance of the proposed algorithm.

This paper is organized as follows: Section 2 provides an overview of the proposed stable and robust object tracking method. Section 3 discusses the geometrically enriched background modelling, while Section 4 presents the iterative method constrained by shape and time properties for estimating confident image regions for background updating. Finally, Section 5 illustrates the visual fall detection metrics, while experimental results are discussed in Section 6. Section 7 concludes the paper.
2. Overview of the Proposed Scheme

The first steps that should be taken in detecting falls based on visual observations is to accurately but also quickly extract the foreground objects (humans) from the visual background. This should be achieved regardless of the complexity and the dynamics of the background. Having detected the foreground object, one can analyse the trajectory of the moving components to decide whether a fall has occurred or not.

To handle any dynamic visual condition in the background, we adopt the following methodology to model background content. Figure 1 shows a typical example of the dynamic nature of our background environment. Figure 1(a) presents four background objects, one fully occluded by the human, two moving and one static. Figure 1(b) depicts the positions of these objects and the human after few seconds. A totally different background condition is noticed where new objects also appear in the scene.

![Figure 1. An example of our dynamic background environment. (a) An image frame at a previous time. (b) An image frame at the current time.](image)

To handle these difficulties, we initially exploit motion information to roughly localize the foreground objects. Parts of the background which are moving are also considered as foreground. First, the motion in the background will be stabilized and the objects will cease to move. Then, these objects will be automatically assigned to the background and correct background subtraction takes place. The same is valid if the foreground stops moving. It will be instantaneously considered as background. However, once the foreground object starts moving again, it will be excluded from the background and correct localization takes place. This is explained in Figure 2. Assuming the background objects are moving within a scene as a result of human force (human activities), it is clear that all moving background objects will asymptotically cease to move and will become part of the background, while sometimes humans start moving and thus will be considered part of the foreground.

To do this, we need (a) a background modelling module which will be able to capture the visual complexity of the background content, (b) a motion detector that will be able to identify coherent motions in a scene in real-time, (c) a foreground approximator which will provide a rough estimation of the contours of the foreground objects, and (d) the fall detection module that give an alerts when falls take place.

3. Modelling the Background through Local Geometry

Traditional methods probabilistically represent the background content by modelling the values of the single pixels of an image through the use of mixture models (e.g., the Gaussian Mixture Models – GMMs). However, such approaches are not reliable in cases where some background regions present similar colour properties with foreground parts. This situation is common in real-life application scenarios; for instance, parts of the person’s clothes have similar colour values with background objects. To overcome this difficulty, we exploit, in this paper, the structure of the local geometry of the pixels as regards background modelling. In particular, we form locally connected structures in order to capture the relations of the pixels within an image area. These pixel relations, which reveal geometric properties of a region, are then used to model the background content instead of single pixel values. In this paper, we have adopted the Gaussian Mixture Models in background modelling for reasons of simplicity and robustness. Section 3.1 briefly describes the Gaussian Mixture Models, while Section 3.2 introduces the new proposed geometric structured GMM framework.

3.1 Gaussian Mixtures Modelling

Let us assume that $M$ Gaussian distributions are sufficient for modelling the background. As we will show in Section 3.3, we incrementally increase the number $M$ so as to find the most suitable number of mixtures that optimally represent the content. We denote the $i$-th Gaussian distribution as $G(p(t), \mu_i(t), \Sigma_i(t))$, where $p(t)$ is the input of the mixture at the time instance $t$. In traditional GMM methods, vector $p(t)$ is the three colour components of a single image pixel. Instead, in this paper, vector $p(t)$ corresponds to the geometric relations of the image pixels within a local region $S$. Detailed description of how vector $p(t)$ is formed is presented in Section 3.2. Variable $\mu_i(t)$ is the mean value of the $i$-th Gaussian at...
time $t_j$, whereas $\Sigma_j(t)$ is a matrix that refers the standard deviation of the multi-variable Gaussian distribution $G(t)$. Then, we model the background content as a mixture of $M$ Gaussian multi-variable distributions such as

$$B(p(t)) = \sum_{i=1}^{M} w_i(t) \cdot G(p(t), \mu_i(t), \Sigma_i(t)) \quad (1)$$

Eq. (1) $w_i(t)$ refers to the coefficient of the $i$-th Gaussian distribution to the background content, $B(t)$ indicates the overall probability an image region, represented by the vector $p(t)$ belongs to background, and $G(t)$ the multi-variable Gaussian distribution given by

$$G(p(t), \mu_i(t), \Sigma_i(t)) = \frac{1}{\sqrt{2\pi}^{d/2} \Sigma_i(t)^{1/2}} \exp\left(-\frac{1}{2} (p(t) - \mu_i(t))^T \Sigma_i(t)^{-1} (p(t) - \mu_i(t))^T\right) \quad (2)$$

In eq. (2) variable $k$ indicates the dimension of the vector $p(t)$, i.e., the dimension of the multi-variable Gaussian distribution.

As new values are labelled as background and/or foreground pixels, the coefficients $w_i(t)$ of eq. (1) should be dynamically updated. Let us assume that a new image region is confidently estimated to belong to the background content. This is achieved, in this paper, based on an innovative algorithm which is discussed in Section 4, and combines motion information fused by shape and time constraints. In particular, let us assume that the algorithm of Section 4 estimates new geometric structures that currently represent background content. These geometric pixel relations are included in a new vector, say $p(t+1)$. Then, the values of the vector $p(t+1)$ are checked against the existing $M$ Gaussian distributions to update the mixture model parameters, the coefficients $w_i(t)$, the mean value $\mu_i(t)$ and the standard deviation matrix $\Sigma_i(t)$. Assuming the independence of the multi-variable components of the Gaussian, the standard deviation matrix $\Sigma_i(t)$ can be written as a diagonal matrix of the form $\Sigma_i(t) = \sigma_i(t) \cdot I$. Regarding the updating procedure, the following different cases can be considered.

**Case 1**: The values of the new vector $p(t+1)$ can be modelled by one of the $M$ available Gaussian distributions, say the $m$-th one. This is achieved if the values of vector $p(t+1)$ are within the interval $[\mu_m - 2.5 \cdot \sigma_m, \mu_m + 2.5 \cdot \sigma_m]$, meaning that we have a tolerance of $\pm 2.5 \cdot \sigma_m$ where $\sigma_m$ is the standard deviation of the $m$-th Gaussian. The respective parameters of the GMM are then updated as follows:

$$w_m(t+1) = w_m(t)$$
$$\mu_m(t+1) = (1 - r) \mu_m(t) + r p_m(t+1)$$
$$\sigma_m^2(t+1) = (1 - r) \sigma_m^2(t) + r (p_m(t+1) - \mu_m(t))^2 (p_m(t+1) - \mu_m(t))$$

where $r$ is given by the following equation:

$$r = \alpha \cdot G(p(t+1), \mu_m(t), \sigma_m(t)) \quad (4)$$

Factor $\alpha$ is a learning rate. The remaining Gaussian distributions are also updated as

$$w_i(t+1) = (1 - \alpha) w_i(t)$$

$$\mu_i(t+1) = \mu_i(t)$$

$$\sigma_i(t+1) = \sigma_i(t)$$

where $i \neq m \quad (5)$

**Case 2**: The values of the vector $p(t+1)$ cannot be modelled by any of the existing $M$ available Gaussian distributions. In this case, a new Gaussian is created to model the statistics of those particular pixel-relations of the region.

**Case 3**: In cases where no new vector $p(t+1)$ is assigned to some of the available $M$ distributions for long periods, the respective distributions are deleted from the mixture. This means that these distributions represented background regions which have currently disappeared from the background content. This is true for example in the case of active cameras where background content has been significantly changed.

### 3.2 Local Geometry in Background Modelling

A simple idea to model the local geometry of an image region is to exploit the colour value components of all the pixels within the region $5 \times 5$ pixels. This way, we expand the traditional $3 \times 1$ pixel vector representation, composed of the three RGB colour components, into a multi-dimensional hyper-space representation, where the respective pixel vector includes all the colour components of the pixel itself and the respective surroundings. For example, in the case of image regions of $3 \times 3$ pixels, we can use a $27 \times 1$ vector (9 structuring pixels x 3 colours = 27 vector elements) for the modelling of this region. Such an expansion compensates the probability that similar image regions simultaneously belong to the background and the foreground; however, it is sensitive to noise and the alteration of the pixel values due to illumination changes. It increases modelling complexity, especially in cases where large image regions are being considered. For example, in the case of an image region of $64 \times 64$ pixels, we conclude a vector $p(t)$ of 12288 ($64 \times 64 \times 3 = 12288$) and thus a similar dimension as regards the multi-variable normal distributions used in the mixture model. Modelling the structure of such extremely high hyperspaces is an arduous task, and practically impossible. We need a very large number of Gaussian distributions, which significantly increases computational complexity, a critical factor for on-line human fall detector architectures. In addition, such an approach is very sensitive to noise, reducing the background modelling performance; small differences in an image region results
in quite different regions of the hyper-space. To address these difficulties, we reduce the dimensionality of the hyper-space using the concept of saliency maps.

3.2.1 Salient-Based Dimensionality Reduction

Let us suppose that we have an image region $S$. According to this methodology, we highlight the most salient locations of the region where the image content is more “informative” in the sense of some criteria. In this way, we create a master (active) map in the image region; we are thus able to select the most important pixels as regards background modelling. Let us denote as $AM(i,j)$ the respective active map. In fact, $AM(i,j)$ is an $s^2 \to R$ mapping, since it takes as input the location of a pixel $(i,j)$ within the image region $S$, and yields as output a scalar value which refers to the importance (saliency) of the pixel $(i,j)$.

In the following, we denote the dissimilarity $D(i,j,n,m)$ between two pixels $(i,j)$ and $(n,m)$ of the image region $S$ [27].

$$D(i,j,n,m) = \log \frac{p(i,j)}{p(n,m)}$$

where we denote as $p(i,j)$ and $p(n,m)$ the pixel values at the locations $(i,j)$ and $(n,m)$. Eq. (6) is a natural definition of dissimilarity. Consider now a fully-connected directed graph $G=\{n, E\}$, obtained by connecting every node of the lattice of the image region, labelled with two indices $(i,j)$ with all other $n$-1 nodes. The node, say $n_{ij}$, of the graph refers to the pixel $(i,j)$ of the image region $S$, while graph edges express the dissimilarity degree between two nodes [see eq. (6)] adjusted by a weight that describes the physical distance (in terms of pixel locations) for the two examined pixels. Then, eq. (6) is modified as

$$E(n_{ij}, n_{im}) = \log \frac{p(i,j)}{p(n,m)} R(i-n, j-m)$$

where $R(i-n, j-m)$ is a decaying factor that expresses the distance between the pixels $n_{ij}, n_{im}$. In our case, the decaying factor $R()$ has the form of

$$R(a,b) = \exp \left( - \frac{a^2 + b^2}{2 \sigma^2} \right)$$

Variable $\sigma$ is a free parameter that determines the effect of the factor $R()$ on the generated graph.

In the following, we define a Markov chain over the constructed graph $G=\{n, E\}$. In this paper, we adopt a normalization procedure for this purpose, so that the maximum value of the graph edges takes a value of 1. The transition probabilities coincide with the normalized edges of the graph. Then, a random walk approach is adopted to estimate the most “informative” pixels of the selected image region. Therefore, the result of this process will be the construction of the Activation Matrix $AM(i,j)$, which is an $s^2 \to R$ mapping that indicates the importance of each pixel in the image region $S$.

As in Section 3.2.A, we assume that we need to select $b$ pixels out of the $s \times s$. Thus, we construct a histogram of the values of the Activation Mask and use the histogram bins as the input of the background models.

4. Estimate of Confident Background Image Regions

To accelerate the time and make the algorithm applicable for real-time, on-line captured video frames, we re-model background regions only in areas that can be highly confidently designated as background. Then, background updating takes place for these areas using the algorithm described in Section 3.

In our approach, estimating these background image regions is performed using an iterative motion detection scheme constrained by shape and time properties. This means that the foreground object is currently detected as a moving object presenting human shape constraints while retaining its continuity in time (temporal coherency).

Our approach assumes that confident background regions are more accurately detected in scenes of high motion activity. This means that motion information is exploited to estimate the likelihood of an image region belonging to the background. To eliminate possible noise in the estimation of the motion field, non-linear filtering methods are applied, tailored by shape-time constraints. As far as the shape constraints are concerned, human objects present aspect ratio limitations in their silhouette. Motion objects that deviate from human shape constraints are excluded from the foreground. We also impose temporal consistency constraints. If a detected motion mask is significantly changed over a short time period, this is an indication of erroneous estimation of the motion field. In such scenarios, we ignore the detected foreground object.

4.1 Iterative Motion Estimation

In this paper, the motion field is detected through an iterative implementation of the Lucas Kanade Optical Flow method [29]. However, motion detection is a noise sensitive process; for instance, a different focus of the camera, which is a usual process of the current camera sensors, may result in estimating large intensity values of motion vectors, though no motion information is encountered in the captured image frames. To overcome these difficulties the constrained shape and time mechanism is proposed as described in Section 4.4.
In particular, let \( F_q(t) \) be the image intensity at the \( q \) pixel of frame \( F(t) \). It is then clear that the motion vector, say \( \mathbf{v} \) of that \( q \) pixel, is given as the solution of the following equation:

\[
F_q(t) = F_{q,l}(t + 1)
\]

(9)

Differentiating eq. (9), we can conclude the following linear equation:

\[
F_{q,x}v_x + F_{q,y}v_y = -F_{q,l}
\]

(10)

In (10), \( F_{q,x} \) and \( F_{q,y} \) are the derivatives of \( F_q \) with respect to the \( x \) and \( y \) location of the \( q \) pixel. \( F_{q,l} \) is the derivative with respect to \( t \). Finally, \( v_x, v_y \) are the \( x \) and \( y \) elements of motion vector \( \mathbf{v} \), i.e., the horizontal and vertical motion, respectively. These derivatives are easily estimated in the discrete image plane using pixel differences. Eq. (10) is a linear equation of two unknowns. Thus, we need an additional constraint to uniquely calculate these two variables [27]. One of most common approaches used is the adoption of coherence; that is, all pixels in a region should have almost the same motion. Let us denote as \( R_x \) and \( R_y \) the rectangular pixel region. We can then conclude \( R_x \times R_y \) linear equations and then, by the pseudo inverse, we can solve this linear system of two unknowns.

\[
\begin{bmatrix}
F_{q,x}v_x + F_{q,y}v_y \\
\vdots
\end{bmatrix}
\begin{bmatrix}
v_x \\
\vdots
\end{bmatrix}
= 
\begin{bmatrix}
-F_{q,l} \\
\vdots
\end{bmatrix}
\Rightarrow
\mathbf{A} \cdot \mathbf{v} = \mathbf{b}
\]

(11)

\( \Rightarrow \mathbf{A} \cdot \mathbf{v} = \mathbf{b} \) for all pixels in region \( R_x \times R_y \)

In (11), matrix \( \mathbf{A} \) is not rectangular and thus for the solution we need to calculate its pseudo inverse:

\[
\mathbf{v} = (\mathbf{A}^T \cdot \mathbf{A})^+ \cdot \mathbf{b}
\]

(12)

4.1.1 Hierarchical Estimation

To accelerate the optical flow algorithm, we adopt an iterative implementation of the algorithm as discussed in [22]. The iterative implementation starts with the creation of a stack of different image resolutions. This is achieved by low-pass filtering of the image content. Let us denote as \( F^{(l)}(t) \) the \( l \)-th resolution decomposition of the image \( F(t) \) at time \( t \). The value of \( l=0 \) corresponds to the highest resolution, i.e., the full image content. Instead, as the value of \( l \) increases gradually, smaller image resolutions are constructed. Given the stack of image frames \( F^{(l)}(t) \) with \( l=0, 1, \ldots \), where \( K+1 \) is the number of decomposed image frames, we first apply the Lucas-Kanade method [27] at the deepest resolution; that is, \( F^{(K)}(t) \). We then propagate the results to the next level \( F^{(K+1)}(t) \). If we also denote as \( \mathbf{v}^{(l)} \) the respective motion vector at the \( l \)-th resolution for the \( q \) pixel, we can relate this motion vector with the vector in the previous resolution \( \mathbf{v}^{(l-1)} \) as

\[
\mathbf{v}^{(l)} = \mathbf{v}^{(l-1)} + \mathbf{r}^{(l-1)}
\]

(13)

where \( \mathbf{r}^{(l-1)} \) indicates the motion vector residual. The motion vector residual can be estimated via the Lucas-Kanade optical flow method [27]. Thus, we can achieve an iterative implementation for motions, which improves the reliability of the method.

4.2 Robust Points for Motion

A critical constraint for our proposed fall detection algorithm is its real-time performance. However, estimation of motion activity over all pixels of an image is in general a time-consuming process which may inhibit the real-time implementation of the algorithm. Additionally, if we apply the aforementioned iterative approach for all pixels of a frame we certainly derive a significant portion of erroneous motion vectors, mainly due to the high dynamics of the background content and the complexity of the visual environment. It is also clear that erroneous estimation of motion vectors results in an erroneous fall detection, increasing false positives/negatives. To address this difficulty, we estimate the motion vectors on particularly selected points on the image plane. This is done by extracting appropriate features from the image pixels and then by selecting “good” pixels using the extracted features.

Several approaches have been presented in the literature for detecting “good” image pixels. In our approach, we implement the method of Shi and Tomasi for pixel representation as feature vectors [26]. We select the Shi and Tomasi method because it exploits the matrix \( \mathbf{A} \) used in estimating the Lucas-Kanade optical flow vectors.

We recall from (11) that motion vector estimation requires the inversion of the matrix \( \mathbf{A}^T \mathbf{A} \). This means that minimum value of the eigenvalues of \( \mathbf{A}^T \mathbf{A} \) should not be equal to zero. Practically, we would prefer the minimum eigenvalue of \( \mathbf{A}^T \mathbf{A} \) to be large enough so that it characterizes “more salient” pixel points in the frame. Therefore, the minimum eigenvalue of \( \mathbf{A}^T \mathbf{A} \) characterizes pixels that are easy to track. For this reason, we estimate the minimum eigenvalue for all pixels in an image and then retain image pixels that have a minimum eigenvalue larger than a given percentage of the total maximum over all pixels.

4.3 Motion Filtering

Another important issue, however, is discarding false motion vectors that may derived either from slight fluctuation of the pixel values due to the camera’s capturing errors, or motions that may be caused by shadows, light reflection or other visual effects. This can
be achieved by taking into account coherence properties as regards the motion vectors. Upon an object’s movement, the detected motion information should have high intensity and almost similar direction; in other words it should be coherent. Thus, we refine the “good” pixels by the property.

More specifically, let us denote as \( \mathbf{v}_R = [v_{R,x}, v_{R,y}]^T \) the motion vectors for all pixels \( q \in R \). We then calculate the intensity of these vectors as \( I = (v_{R,x})^2 + (v_{R,y})^2 \) and the direction angle as \( \theta = \arccos(v_{R,y}/v_{R,x}) \). It is clear that we prefer pixels of high intensity motion vectors (high activity). We also prefer vectors of coherent motion, i.e., of almost similar angles \( \theta \). The first fact can be easily formulated by thresholding the intensity values. The second fact can be obtained by calculating the standard deviation of the angles \( \theta \) for all pixels within image regions and retaining the ones whose values is small, i.e., they present almost similar directions. It is clear that alterations in lighting conditions or shadows can cause no coherent motion, which is detected by the filtering algorithm as a false motion. In these cases, the background does not update, reducing tracking efficiency, especially when the background content has undergone a significant modification (e.g., high fluctuations in background lighting conditions). However, this normally lasts only a few image frames. The lighting conditions are stabilized quickly (lighting alterations do not often last for many image frames) leading to an estimate of coherent motion regions for the correct following frames and thus to a new background updating. This, consequently leads to a self-correction of the tracking performance (see Section 6 and Figure 3).

4.4 Constrained Shape and Time Fusion

In this section, we present the algorithm that we have used to detect the foreground object by exploiting motion information (see Sections 4.1-4.3). Once the foreground object has been detected, we can estimate the most confident background image regions, which are then used for updating the parameters of the background model based on the local geometrically enriched framework, as discussed in Section 3.2. The foreground object is detected through the motion estimation algorithm described above. In particular, the following scenarios are discriminated:

**Case #1: No significant motion information** In cases where no significant motion information is encountered in the examined image frame (e.g., there are not enough motion vectors after the filtering procedure), no foreground object is detected and therefore no background regions. In this scenario, the background model remains unchanged (no activation of the updating procedure).

**Case #2: Significant motion information** In cases where significant motion detection in the image scene is encountered, we are able to detect a foreground object around the most important detected motion information as derived after the filtering approach. The detected foreground masks then undergo validation of shape and time constraints, as presented in the following.

4.2.1. Shape Constrained fusion The first rule considers the shape properties of the detected foreground object. The detected foreground mask should satisfy the shape constraints of a human object. This means that the aspect ratio of the detected foreground region should satisfy the aspect ratio of a human object. Otherwise, we assume that the foreground has not been accurately detected; we therefore do not update the background model. In our experiments, we have chosen the aspect ratio to be within the range of \([0.2 - 0.25]\). It is clear, however, that due to occlusions it is probable the detected aspect ratio is significantly changed from the aforementioned values. This is depicted in Figure 5, where the detected foreground object comes close enough to the camera. Shape constraints are not satisfied and thus the background is not updated. However, this does not lead to a deterioration of the tracking performance (and consequently to a deterioration of the detected fall) since the background content does not significantly change. An example of this situation is discussed in Figure 5.

4.2.2. Time Constrained fusion A human movement lasts several image frames. Thus, we need to evaluate the continuity of the detected regions overtime. This means that the intersection between two detected foreground masks at two consecutive frames should be small in order to guarantee temporal coherence of the human object. In cases where there is an abrupt shape modification of the detecting foreground objects, this would probably come from a noise condition. The algorithm therefore assumes that the detected foreground mask is not reliable, and the complimentary images cannot be used for updating the background model.

5. Fall Detection

In the previous sections, we describe an efficient algorithm for foreground extraction with dynamic background conditions. We exploit the foreground segmentation for efficiently detecting a person’s fall from visual cues, which is a very important research aspect in the computer vision society. In particular, we procure measurements over the extracted mask of the foreground object in order to determine whether a fall has been occurred or not.

One simple approach estimates the centroid vertical velocity in a similar way to [11]:

```
where we denote as $V^{(c)}$ the velocity of the centroid of the foreground mask and as $s^{(c)}$ the $x$ and $y$ coordinates of the respective centroid. In eq.(14), we approximate the velocity as the difference of the centre of the mask over the image frames $k$ and $k-1$.

The main drawback of such an approach is that the centroid of the mask is not so accurate as to distinguish normal human activities (such as sitting) from the fall. To address this difficulty we use the upper bound of the foreground mask instead of the centroid. Thus, we substitute $s^{(c)}$ with $s^{(u)}$ in eq. (14), resulting in the calculation of the $V^{(u)}$ instead of $V^{(c)}$.

Again, however, such an approach presents several shortcomings. First, in cases where the detected foreground mask has been erroneously estimated, the velocity $V^{(u)}$, calculated as a difference operator, enhances the noise. Therefore, there are a lot of errors in the detection of a fall using the metric of $V^{(u)}$.

To deal with this issue, we estimate the accumulation of the velocity over several frames, resulting in a metric of the form $E[V^{(u)}]$ where $E[\cdot]$ refers to the expectation operator. Again, however, a velocity-based metric presents the drawback that it is sensitive to different orientations of the fall (towards the camera, in an opposite direction or even perpendicular to the camera). For this reason, a new metric is introduced in this paper that is composed of a normalized version as regards vertical velocity constrained over a ratio of the height of the detected foreground mask. Normalization of the vertical velocity faces the problem of scaling as regards human object capturing. Therefore, it is not the absolute difference of the vertical velocity that determines whether a fall has occurred, but the percentage with respect to the initial position of the foreground mask. Additionally, the constrained weight of the height ratio enhances the normalized vertical velocity or it does not, according to the scale of the detected mask:

$$ NV^{(u)} = \frac{t}{i-t-k} \sum_{i=t-k}^{t} \frac{s_y(i) - s_y(i-1)}{s_y(i)} \frac{h(t)}{h(t-k)} $$

where $h(\cdot)$ indicates the height of the detected foreground mask.

We have introduced an additional criterion for detecting a fall. The criterion stems from the fact that the estimated motion vectors have quite different distribution over a fall or over a normal activity.

$$ OFM = \text{std}(\theta_{\text{det}}, \theta_{\text{normal}}) + \lambda \cdot |v| $$

where we denote as $v$ the respective vectors derived from the optical flow algorithm and as $\theta$ the respective angles of the optical flow. The OFM refers to the optical flow metric. To detect a fall, both criteria are taken into account combined through the ‘AND’ operator. A fall is then detected through the criterion

$$ NV^{(u)} + \gamma \cdot OFM $$

where $\gamma$ is a constant scalar variable.

6. Experimental Results

6.1 Description of the Evaluation Environment

The proposed visual fall detection service has been evaluated in real-life experiments (during the activities of the Intelligent System for Independent Living and Self Care of Seniors with Cognitive Problems or Mild Dementia project [25]). The visual fall detection system of ISISEMD has been developed in C platform and by exploiting the Integrated Performance Primitives (IPP) of Intel [29] [30], in order to support real-time execution of the aforementioned algorithms. The developed interface allows MMS service to the caregivers; if a fall is detected, the system sends three image frames to caregivers for assessing the situation. Socket interfaces have been used for communication of the visual fall detection service with the additional ISISEMD services. Before the real-life installation of the ISISEMD visual fall detection service, the system has been evaluated in the laboratory for the accuracy of both foreground detection and fall.

6.2 Tracking Evaluation.

We have applied a list of evaluation scenarios to test our approach. Table 1 presents the evaluation scenarios used along with the tracking efficiency achieved by the proposed algorithm. Additionally, we have depicted the number of frames required for the algorithm to be converged with the new conditions.
Figure 3. Tracking results and the effect of alterations in lighting conditions in tracking. We depict optical flow motion vectors at the good-to-track points [29] over the original images (the first row) and the respective foreground mask after morphological filtering (the second row).

Figure 4. The effect of an active camera on the tracking performance. It seems that after a few frames the algorithm converges; first row: different visual changes along with the detected optical flow vectors; second row: the estimated foreground mask.

Figure 5. Tracking performance in cases where the shape constraints are changed; in frames when the shape constraints are violated, the background is not updated. However, this does not lead to a deterioration of the tracking, since no significant background changes are encountered. In the next frames, the shape constraints are again satisfied and the background is updated.

Figure 3 presents tracking results of the proposed dynamic background modelling algorithm through the exploitation of local geometry. It also presents the optical flow motion vectors, estimated at the good-to-track points [26]. Figure 3 illustrates the effect of lighting conditions in tracking performance by presenting five key frames from a scene. With a significant alteration of the lighting conditions (some of the lights of the room are turned off), no coherent motion is detected and thus the motion filtering component triggers no background updating (see Section 4.3 for more details). This deteriorates tracking, since the background pixels have been changed due to luminosity changes. However, in the next few image frames the lighting conditions are stabilized, leading to an estimation of coherent motion information and thus to an improvement of the tracker. Similarly, in Figure 4, we have depicted the effect of an active camera on tracking performance. With severe visual changes, the tracker does not need many frames to converge with the new visual conditions (see Table 1). This is accomplished through the background model updating procedure (see Section 4). Similarly, Figure 5 presents the tracking performance where the shape constraints of the fusion component (see Section 4.4) are violated. As is observed, tracking still remains accurate although no background updating is encountered, since the background content remains almost the same.

The ability of the tracker to rapidly converge with the new visual conditions reveals the stability of the proposed tracking scheme. Table 1, appears to show that, usually, 25 frames are enough to converge the tracker. Even in the worst case of a complete visual change (due to an active camera effect) up to 100 frames (about 4 seconds) are adequate for the convergence.
Table 1. The evaluation scenarios used along with the tracking accuracy and the time (measured in terms of # of frames) needed for the algorithm to converge.

| Evaluation Scenarios                         | Tracking Accuracy Achieved (%) | Average algorithm convergence (total frames) |
|---------------------------------------------|--------------------------------|-------------------------------------------|
| 1. Luminosity change (the lights of the room are turned on/off, the sun light significantly changes) | >85%                           | 5 frames                                  |
| 2. New foreground enters/leaves the scene    | >70%                           | 24 frames                                 |
| 3. Background changes                       | >80%                           | 19 frames                                 |
| 4. Active Camera (camera rotation, translation, different focus, zoom in/out) | >60%                           | 100 frames                                |

Figure 6. The performance of the proposed object tracking algorithm that exploits local geometric and self adaptability in comparison with other approaches.

Figure 6 illustrates the performance of the proposed object tracking algorithm versus number of frames. The results have been obtained using a collage of four segments, each corresponding to different visual environmental conditions such as the ones in Table 1. This collage is constructed to validate the robustness and adaptivity of the proposed tracking algorithm to abrupt visual changes. It also depicts tracking performance of non-adaptive background models, as well as the model where no geometrically enriched local structures are used. As is observed, the proposed methodology outperforms the compared ones.

6.3 Fall Detector Evaluation.

Figure 7 presents the different distribution of the optical flow when a fall is detected compared to other normal human activities. Therefore, in cases where we combine the normalized vertical velocity with the fluctuation of the estimated motion vectors, we are more confident in detecting a fall.

Table 2 summarizes the sensitivity and specificity metric derived from False Positive-FP (a fall that is wrongly detected), False Negative-FN (a fall occurred but it is not detected), True Positive-TP (a fall occurred and detected) and True Negative-TN (a normal human activity occurred and was detected). In particular, sensitivity is the ratio of TP/(TP+FN) while specificity is the ratio of TN/(FN+TP). The results in Table 2 have been yielded by a real-life operation of the system in the municipality of Trikala. In the system operation more than 5000 normal human activities have been tested and more than 170 falls have been examined [25]. The results have been obtained by initially calibrating the system to an Equal Error Rate Operation. This is achieved by adjusting the threshold using a set of 20 initial falls and normal human activities. The system then operates with the set of the initial estimate parameters. Additionally, Table 3 presents how the fall positive error is distributed across different human activities.

| Algorithm Adopted                                      | Sensitivity | Specificity |
|---------------------------------------------------------|-------------|-------------|
| Proposed Algorithm (both criteria – normalized vertical velocity and optical flow metric) | 92%         | 93%         |
| Proposed Algorithm (normalized vertical velocity)      | 89%         | 85%         |
| Asynchronous Temporal Contrast [11]                    | 84%         | 81%         |
| LVQ Modelling [23]                                     | 87%         | 77%         |
| Support Vector Machine [18]                            | 85%         | 83%         |
| Shape Deformation [20]                                 | 90%         | 90%         |

Table 2. Comparison of the proposed approach with other methods reported in the literature in terms of detecting falls.

| Human Activities                                    | Sitting | Walking | Bending | Other |
|------------------------------------------------------|---------|---------|---------|-------|
| Proposed Algorithm (both criteria – normalized vertical velocity and optical flow metric) | -4%     | <1%     | <1%     | 3%    |
| Proposed Algorithm (normalized vertical velocity)   | -9%     | -2%     | -2%     | -2%   |
| Asynchronous Temporal Contrast [11]                 | -11%    | -1%     | -5%     | -2%   |
| LVQ Modelling [23]                                   | -13%    | -4%     | -3%     | -3%   |
| Support Vector Machine [18]                         | -12%    | -3%     | -2%     | -1%   |
| Shape Deformation [20]                               | -7%     | <1%     | -1%     | -2%   |

Table 3. Distribution of the fall positive error across different human activities.
7. Conclusions

This paper presents an innovative fall detection algorithm based on visual cues that exploits robust and stable foreground object tracking methods. The proposed tracking approach combines local geometric structures used for background modelling and iterative motion information, constrained by shape and time properties for background content updating. This way, we achieve a continuous foreground monitoring taking into account visual changes like luminosity fluctuations, active cameras, background/foreground content changes, partial/full occlusions, and appearance/disappearance of new objects.

Our method has been tested in real-life conditions and tested against complex daily human activities like sitting, walking and bending. The computational complexity of the developed scheme makes it implementable in real-time over long periods. Finally, its superior performance was tested in real-world scenarios.

8. Acknowledgments

This research has been supported by European Union funds and national funds from Greece and Cyprus under the project “POSEIDON: Development of an Intelligent System for Coast Monitoring using Camera Arrays and Sensor Networks” in the context of the inter-regional programme INTERREG (Greece-Cyprus cooperation), contract agreement K1 3 10–17/6/2011.

9. References

[1] Murray C.J.L and Lopez A.D (1996) Global and regional descriptive epidemiology of disability: Incidence, prevalence, health expectancies and years lived with disability. Global Burden Dis., 1:201–246.
[2] Masud T, and Morris R.O (2001) Epidemiology of falls. Age Ageing, 30:3–7.
[3] Thome N, Mignet S, Ambellouis S (2008) A real-time multiview fall detection system: A LHMM-based approach. IEEE Trans. on CVST, 18:1522–1532.
[4] Bourke K and Lyons G.M (2008) A threshold-based fall-detection algorithm using a bi-axial gyroscope sensor. Med. Eng. Phys., 30:89–90.
[5] Lai C.F, Chang S.Y, Chao H.C, Huang Y.M (2011) Detection of Cognitive Injured Body Region Using Multiple Triaxial Accelerometers for Elderly Falling. IEEE Sensors J., 11:763–770.
[6] Bianchi F, Redmond S.J, Narayanan M.R, Cerutti S and Lovell N.H (2010) Barometric Pressure and Triaxial Accelerometry-Based Falls Event Detection. IEEE Trans. on Neural Systems and Rehabilitation Engineering, 18:619–627.
[7] Srinivasan S, Han J, Lal D, and Gacic A (2007) Towards automatic detection of falls using wireless sensors. Proc. 29th Annu. Int. Conf. IEEE EMBS, 1379–1382p.
[8] Barrett A, O’Connor M, Culhane K, Finucane A.M, Olaighin G, Lyons D (2008) A footswitch evaluation of the gait of elderly fallers with and without a diagnosis of orthostatic hypotension and healthy elderly controls. 30th Annu. Int. Conf. IEEE EMBS, 5101–5104p.
[9] Litvak D, Gannot I, and Zigel Y (2008) Detection of falls at home using floor vibrations and sound. IEEE 25th Conv. Electr. Electron. Eng, 514–518p.
[10] Tseng H.W, Chen M.Y, Chen J.Y (2010) Design of fall detection system with floor pressure and infrared image. Inter. Conf. on System Science and Engineering, 131–135p.
[11] Fu Z, Delbruck T, Lichtsteiner P, Culurciello E (2008) An Address-Event Fall Detector for Assisted Living Applications. IEEE Trans. on Biomedical Circuits and Systems, 2:88–96.
[12] Lin C.W, Ling Z.H, Chang Y.C, Kuo C. J (2005) Compressed-domain Fall Incident Detection for Intelligent Home Surveillance. IEEE Inter. ISCAS, 3781–3784p.
[13] Liao T, Huang C.L (2010) Slip and Fall Events Detection by Analyzing the Integrated Spatiotemporal Energy Map. 20th International Conference on Pattern Recogn. (ICPR), 1718–1721p.
[14] Diraco G, Leone A, Siciliano P (2001) An active vision system for fall detection and posture recognition in elderly healthcare. Design, Automation & Test in Europe Conf. & Exhibition (DATE), 1536–1541p.
[15] Auvinet E, Multon F, Saint-Arnaud A, Rousseau J, Meunier J (2011) Fall Detection With Multiple Cameras: An Occlusion-Resistant Method Based on 3-D Silhouette Vertical Distribution. IEEE Trans. on Inform. on Technology in Biomedicine, 15:290–300.
[16] Doukas C.N, Maglogiannis I (2011) Emergency Fall Incidents Detection in Assisted Living Environments Utilizing Motion, Sound, and Visual Perceptual Components. IEEE Trans. on Information Technology in Biomedicine, 15:277–289.
[17] Lai C.F, Huang Y.M, Park J.Y, Chao H.C (2010) Adaptive Body Posture Analysis for Elderly-Falling Detection with Multisensors. IEEE Intel. Systems, 25:20–30.
[18] Foroughi H, Rezvanian A, Pazirae A (2008) Robust Fall Detection Using Human Shape and Multi-class Support Vector Machine. Computer Vision, Graphics & Image Processing, 413–420p.
[19] Jamil Khan M, and Adnan Habib H (2009) Video Analytic for Fall Detection from Shape Features and Motion Gradients. Proc. of the World Congress on Engineering and Computer Science USA, 1311-1316p.
[20] Rougier C, Meunier J, St-Arnaud A, and Rousseau J, (2011) Robust Video Surveillance for Fall Detection Based on Human Shape Deformation. IEEE Trans. on Circ. & Syst. for Video Techn., 21:611–622.
[21] Luštrek M, and Kaluža B (2008) Fall Detection and Activity Recognition with Machine Learning. Informatica, 33:205–212.

[22] Doulamis N (2010) Iterative motion estimation constrained by time and shape for detecting persons’ falls. ACM 3rd Inter. conf. on Pervasive Technologies Related to Assistive Environments, 1-8p.

[23] Doulamis A, Doulamis N, Kalisperakis I, and Stentoumis C (2010) A Real-Time Single-Camera Approach For Automatic Fall Detection. ISPRS Commission V, Mid-Term Symposium Close Range Image Measurement Techniques, UK, 207-212p.

[24] Doulamis A (2010) Dynamic Tracking Re-Adjustment: A Method for Automatic Tracking Recovery in Complex Visual Environments. Multimedia Tools and Applications, 50:49–73.

[25] Doulamis A, Kalisperakis I, and Stentoumis C, and Matsatsinis N (2010) Self Adaptive Background Modeling for Identifying Persons’ Falls. IEEE Semantic Media Adaptation and Personalization.

[26] Shi J, Tomasi C (1994) Good Features to Track. Intern. Conf. on Comp. Vis. & Patt. Rec, 593–600p. Harel J, Koch C, Perona P (2007) Graph-based visual saliency. Advances in Neural Inform. Proc. Systems, 593-600p.

[27] Lucas D, Kanade T (1981) An Iterative Image Registration Technique with an Application to Stereo Vision. Imaging Understanding, 121–130p. Integrated Performance Primitives. Available: http://software.intel.com/en-us/articles/intel-ipp/

[28] Doulamis N (2010) Visual Fall Alert Service in Low Computational Power Device to Assist Persons with Dementia. IEEE 4th Intern. Symp. on Applied Sciences in Biomedical and Comm. Techn, 1-5p.

[29] ISISEMD European Union Funded project. Available: http://ec.europa.eu/information_society/apps/projects/factsheet/index.cfm?project_ref=238914