Expert Systems with Applications 39 (2012) 5935–5945

Contents lists available at SciVerse ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Eigenspace-based fall detection and activity recognition from motion templates and machine learning

David Nicholas Olivieri *, Iván Gómez Conde, Xosé Antón Vila Sobrino

Computer Science and Engineering, University of Vigo, Spain

ARTICLE INFO

Keywords:
Human fall detection
Human motion analysis
PCA
Eigenspace classification

ABSTRACT

Automatic recognition of anomalous human activities and falls in an indoor setting from video sequences could be an enabling technology for low-cost, home-based health care systems. Detection systems based upon intelligent computer vision software can greatly reduce the costs and inconveniences associated with sensor based systems. In this paper, we propose such a software based upon a spatio-temporal motion representation, called Motion Vector Flow Instance (MVFI) templates, that capture relevant velocity information by extracting the dense optical flow from video sequences of human actions. Automatic recognition is achieved by first projecting each human action video sequence, consisting of approximately 100 images, into a canonical eigenspace, and then performing supervised learning to train multiple actions from a large video database. We show that our representation together with a canonical transformation with PCA and LDA of image sequences provides excellent action discrimination. We also demonstrate that by including both the magnitude and direction of the velocity into the MVFI, sequences with abrupt velocities, such as falls, can be distinguished from other daily human action with both high accuracy and computational efficiency. As an added benefit, we demonstrate that, once trained, our method for detecting falls is robust and we can attain real-time performance.

1. Introduction

Automatically determining human actions and gestures from videos or from real-time surveillance cameras has received considerable attention both in the academic literature and in commercial applications. Intelligent surveillance systems for the health care industry are particularly attractive since they promise to increase the quality of remote care as well as reduce the growing costs of present remote care methods. Indeed, due to the marked increase in the percentage of elderly persons compared to that of working age population, intelligent home surveillance systems and applications will play an important role in future personalized care systems. For the elderly, video based monitoring could provide a convenient and comprehensive detection system for anomalous behavior, such as falls or excessive inactivity.

In general, determining human motion is a difficult problem in computer vision, and there are many different approaches, including tracking the full 3D body motion with multiple cameras, to Bayesian inference tracking. A recent review by Poppe (2010), provides an updated account of several of the most successful methods. For obtaining information about more limited human motions, such as anomalous activities and falls, the full 3D tracking produces an overabundance of information at the cost of huge computation. Indeed, a more simplistic and computationally viable approach can be found from work on human gait characterization, where dimensionality reduction transforms a sequence of images into points within a canonical space for the purpose of distinguishing types of human gaits by Huang, Harris, and Nixon (1999a) and disorders such as degrees of Parkinson by Cho, Chao, Lin, and Chen (2009). More recently, other authors have described fall detection systems based upon video sequences (see Liu, Lee, & Lin, 2010) using similar techniques.

This paper describes a computer vision software system and algorithms for the detection of human activity using a canonical eigenspace transformation of a novel spatio-temporal motion templates. Machine learning algorithms are applied for discerning the following common activities: walking, walking exaggerated, jogging, bending over, lying down, and falling. We show that fall detection can be accomplished with considerable accuracy without the use of sensors nor a full reconstruction of the 3D human posture. Thus, it is an effective and inexpensive method that can be implemented for real-time monitoring.

In particular, we introduce a new representation, denoted MVFI (Motion Vector Flow Instances) templates, which together with eigenspace methods, provide robust detection for a wide class of indoor human motions. The MVFI template encodes the both the

* Corresponding author. Address: Universidad de Vigo, E.S.E. Informática, 32004 Ourense, Spain. Tel.: +34 988387026.

E-mail addresses: olivieri@ei.uvigo.es (D.N. Olivieri), ivangconde@uvigo.es (I. Gómez Conde), anton@uvigo.es (X.A. Vila Sobrino).

0957-4174/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2011.11.109

size and direction of the optical flow vector from each frame of a motion sequence. Instead of coloring an entire block the same color (as suggested by the MFH (motion flow history) in Venkatesh Babu & Ramakrishnan (2004), we represent the size of the box in *x* and *y* directions independently. This allows us to distinguish vertical and horizontal movements with high precision, which is what is needed for a fall detection algorithm. In our method, the MVFI templates are extracted and projected into a canonical Eigenspace. The projected image templates are used to train LDA classifiers for recognizing a set of six human actions. The technique works well because it is specifically sensitive to large horizontal and vertical velocities, as encountered in falls.

2. Related work

Motivation for fall detection can be found in recent work by Larson & Bergmann (2008), who described the etiology of falls in the elderly. The health risk assessment of falls in the elderly has been studied by Moylan & Binder (2007), and provides ample statistics for demonstrating the seriousness of falls as a major health risk. For example, nearly one third of people over 65 years of age fall each year, and of those 10–15% result in serious injury. More important, 75% of those with fractures do not regain their previous level of activity, constituting a serious quality of life altering hazard.

There are several approaches reported in the academic literature for detecting falls using technologies that have been adapted for home monitoring. In order to build robust fall detection systems for a home environment, however, it is important to understand the characteristics of a typical fall, which has been described by several researchers. In particular, Wu (2000) obtained velocity characteristics of a fall from normal activities, and used this information to build a detection system. In their system, they used body markers and a camera for detecting markers, which allowed them to identify unique features of body movement during a fall incident. From this information, they obtained time trajectories and vertical velocity of the markers. By measuring daily activities, such as walking down stairs, object pickup, getting in/out of tub, and lying down, they were able to quantify the fact that falls (forward, backward, and tripping) produce horizontal/vertical velocities increase from two to three times that produced by normal activity.

Later work from Bourke, O'Donovan, & OLaighin (2007) used human body markers and special video hardware for obtaining extensive velocity profile data on fall events. Experiments were carefully prepared so that subjects would fall into a 0.75 m thick mats, thus making the fall dynamics as realistic as possible without harm to the subjects. They determined a vertical velocity threshold from the experimental profiles, that are sufficient to detect falls with sensors. Sensor based systems are also described in detail in the work of Noury et al. (2007), where they have studied the principles of fall detection and several options with sensors that can be used in practical situations. In order to better characterize the falls, Nyan, Tay, Tan, & Seah (2006) performed measurements and analysis of falls using angular velocity characteristics from high-speed cameras. Nonetheless, these systems predominantly use sensors that the user must wear constantly, which can be cumbersome and prone to disuse.

Detection based purely upon real-time video surveillance has been described in work by Chen, Bharucha, & Wactlar (2007). More complex depth fields were studied by Williams, Ganesan, & Hanson (2007) by considering distributed smart camera network for detecting falls in the home. Recently, Liu et al. (2010) use Zernike polynomials, which capture angular components of motion as multipole expansions, and k-nearest neighbor classifiers to contrast fall events from normal daily activity.

Determining human action in video scenes has a long history within the computer vision community, and has matured enough to have warranted several recent survey articles. Forsyth, Arikan, Ikemoto, O'Brien, & Ramanan (2006) have described kinematic tracking of the human body, and efforts to understand motion. More recently, an extensive survey by Poppe (2010) describes the most successful approaches for labeling image sequences of human actions.

In order to be able to fully describe the human motion, several systems have been developed that effectively *lift* 2-dimensional tracking into a 3-dimensions. Work by Deutscher & Reid (2005), for example, describes the use of stick models with 26 movable angles and using particle filters at each node in order to reconstruct the full human motion. Classic work on activity recognition was described by Bobick & Davis (1996) through appearance based representations called motion templates. They introduced the MHI (motion history instance) and MEI (motion energy instance) templates that attempt to capture information over many image sequences. Venkatesh Babu & Ramakrishnan (2004) formed feature vectors from the same motion templates and showed that several human actions in video sequences could be classified with high precision. Modern methods, described by Meeds, Ross, Zemel, & Roweis (2008) use Bayesian graphical models to infer connected stick models of human and animal motion without imposing prior knowledge of the type of movement. Felzenszwalb, Girshick, McAllester, & Ramanan (2010) show how multi-scale deformable parts models can be used for extracting detailed motions from videos without prior object models. İkizler & Forsyth (2008) have shown that *atomic* three-dimensional movements of body parts, when combined with Hidden Markov models (HMMs), can be used to infer composite motions, much like phonemes are combined to form words. Other notable approaches to characterizing human actions include Han, Wu, Liang, Hou, & Jia (2010), who use hierarchical manifold space and Kellokumpu, Zhao, & Pietikäinen (2010), who have used dynamic textures.

While the goal of these methods is to provide a complete description of human motion, they are computationally expensive for real-time applications. For the present case of fall detection, this is especially true since the objective is to discriminate types of movements. In this limited case, techniques which rely upon spatio-temporal information are better suited and more efficient. The spatio-temporal information can be encoded as a motion template representation, that captures characteristics of movement across frames, either in the form of position or velocity. When combined with dimensionality reduction algorithms, such as PCA, these methods are able to classify different types of movements. The idea has its origins from the work of Etemad & Chellappa (1997) on eigen-faces, who adopted linear discriminant analysis (LDA), also called canonical analysis, to optimize the separability of different face classes. The fundamental insight of this method is that while the essential features of a face are the same, it is covariance which plays an important role in discrimination. Applying this idea of eigenspace transforms to that of human movement was described first for discriminating people based upon their gait. An early account of using eigenspace transforms for analyzing gait was provided by Murase & Sakai (1996), who showed the importance of spatio-temporal correlation for discriminating different types of movements. Huang, Harris, & Nixon (1999b), Huang et al. (1999a) used PCA for classifying gaits of different persons. They introduce the idea of a transform into a canonical eigenspace, consisting first of a PCA followed by a Fisher LDA for reducing the in-class variance while maximizing the out of class variance. Das, Wilson, Lazarewicz, & Finkel (2006) used a two stage PCA which in the first pass reduces the dimensionality, in the second pass the curves are compared in the new space. In an interesting application of gait recognition, Cho et al. (2009) have shown the use of

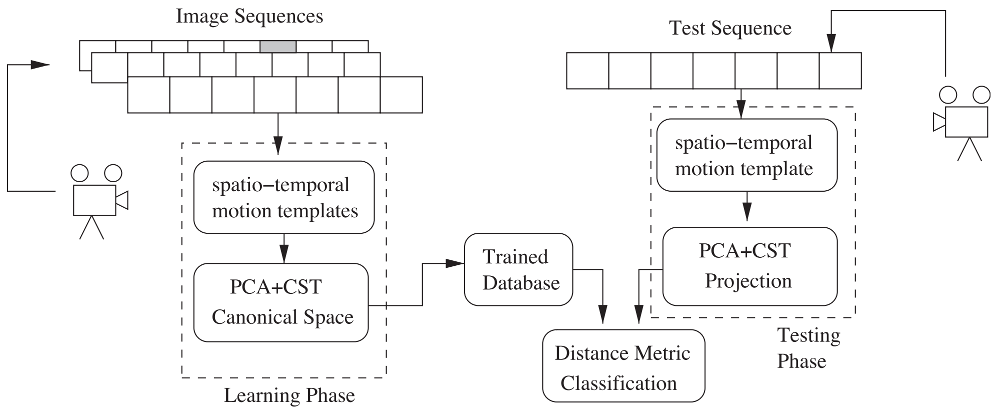


Fig. 1. Workflow for training and testing the fall/action detector. The learning phase converts a set of image sequences pertaining to different action classes to spatio-temporal motion templates, which are subsequently mapped to a canonical eigenspace. A test sequence is transformed to the same eigenspace and a distance metric can be used for classification.

the PCA classifier for analyzing the degree Parkinson disease in elderly patients based upon their gait.

As described, the PCA eigenspace method is based upon motion templates, which have been studied extensively. A motion template can be the silhouette, obtained from foreground/background subtraction (frame differencing), or can be based upon time averaged optical flow. Davis & Bobick (1997) and Bobick & Davis (2001) used motion-energy images (MEI) and motion-history images (MHI); these are binary cumulative motion images and temporal history motion images. In the work of Huang et al. (1999b), & Huang et al. (1999a), MHI and MEI are used. Venkatesh Babu & Ramakrishnan (2004) use the idea of MHI and MFH (Motion flow history). The features extracted from MHI and MFH were used to train KNN, Bayes, Neural Network and SVM classifiers for recognizing a set of seven human actions. More recent variations of the motion templates are described in Ahad, Tan, Kim, & Ishikawa (2009) (temporal motion recognition) and Ahmad & Lee (2010) (variable silhouette). These recent papers underscore the interest in the eigenspace method as well as the fact that *template engineering* together with classification methods can greatly improve the overall recognition results.

3. Theory and algorithms

The PCA based eigenspace method we have used consist of several coordinated steps in order to train our system for automatically detecting falls and actions from video sequences. First a set of canonical transformations (consisting of PCA and LDA) dramatically reduces the dimension of a sequence of images to a set of points in a multidimensional space, albeit significantly reduced in size. The principle workflow of the system is shown in Fig. 1. Supervised learning consists of training the system with many video sequences, from different actions and with different subjects, converted to a motion template sequences which are subsequently transformed to points in the canonical space. An image sequence produces a trajectory in the canonical space, and therefore each trained KNN-based action can be represented by the mean and variance of a large number of such trajectories in the canonical space. Thus, the recognition of an action, from a video sequence is accomplished by forming a motion template, transforming the sequence to the canonical eigenspace, and subsequently selecting the minimum distance of the full trajectory from all the trained action class trajectories.

There are many details of our system, associated with each of the steps of Fig. 1. In particular, we shall describe in this section the preprocessing of video sequences, the construction of the motion templates, the mathematics of the canonical transformation, and the process of machine learning.

3.1. Preprocessing

For this paper, we used the system described in Fig. 1 for classifying six different human actions, with the specific goal of clearly distinguishing fall events. Fig. 2 shows the different actions that we used to train our system. These actions are: walking normal, walking exaggerated, jogging, bending over, lying down, and falling.

For this work, we constructed our own database of video scenes, which we have made available on our web site.[1] Our video database consists of 394 video sequences representing six human actions and using 12 different human subjects. We used a Sony (HDR-HC15) MiniDV filmed with digital resolution, with a sampling rate 25 frames/s. All actions were recorded using the same focal distance and no special back-lighting preparations were implemented, which is of particular importance for shadow effects that the low level motion algorithms needed to eliminate. Actions were performed at approximately the same speed, since the subjects entered the scene one after another.

The videos were saved in AVI MPEG encoding format, and cropped using `ffmpeg` in order to produce image frames $\approx 600 \times 400$. We performed low-level image processing in order to generate silhouettes and spatio-temporal motion templates. All the low-level algorithms we used were written with python bindings to OpenCV2.1 by Bradski (2000), a standard and powerful open source computer vision library. Our low level image algorithms directly process these videos and produce silhouettes and spatio-temporal motion templates images sequences. The individual image frames from the motion template video sequences are extracted, reduced in size to 128×128 using an adaptive resizing algorithm, and converted to gray scale for subsequent transformation into the canonical eigenspace.

3.2. Spatio-temporal motion templates

There are 3 types of motion templates that are used in this paper: (a) silhouettes, (b) MHI templates, and (c) MVFI, which is a novel spatio-temporal template introduced in this paper. We shall describe each of the algorithm and show that the MVFI is especially suited for the detection of falls, that are characterized by abrupt velocity profiles.

The algorithm we used for obtaining silhouettes (of the foreground object) was developed using the robust Gaussian mixture

[1] http://www.milegroup.net/.

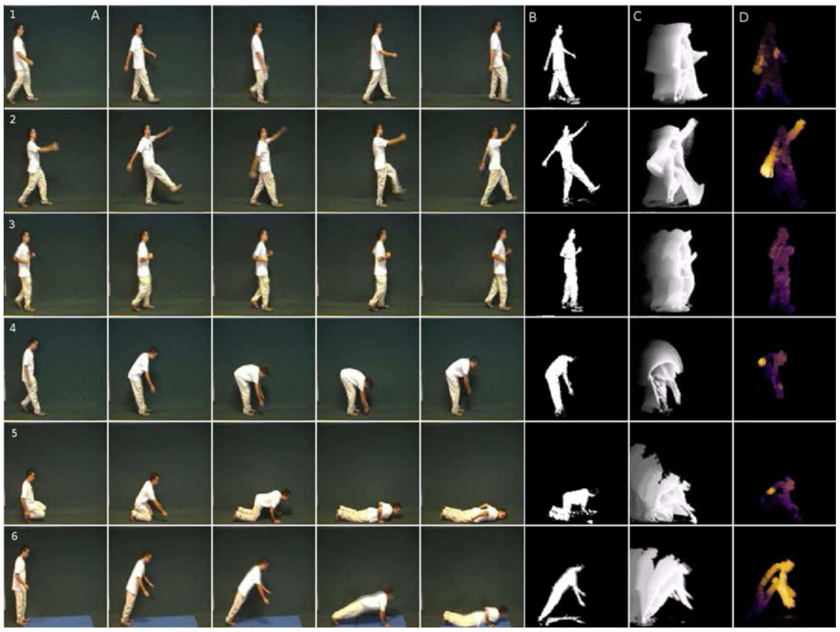


Fig. 2. Actions in our database to be classified/detected. The actions $A1-A6$ are the video sequences for: walking, exaggerated walking, jogging, bending over, lying down, and falling. To the right are example images from the different templates used in this work: $B1-B6$ are silhouettes, $C1-C6$ are MHI motion template, and $D1-D6$ are MVFI motion templates.

model based background subtraction technique by Kaewtrakulpong & Bowden (2001), and later improved by Li, Huang, Gu, & Tian (2003), for detecting foreground objects in complex scenes, and finally by Leone & Distante (2007) which uses a texture analysis in order to improve the performance, particularly for eliminating background shadows. We used the implementation of this algorithm provided by OpenCV2.1 and found that while sensitive to input parameters, the algorithm performs extremely well at eliminating background shadows. In our implementation we also use low-level morphologic operations and thresholding to further reduce background noise in order to produce a crisp silhouette of the human subject.

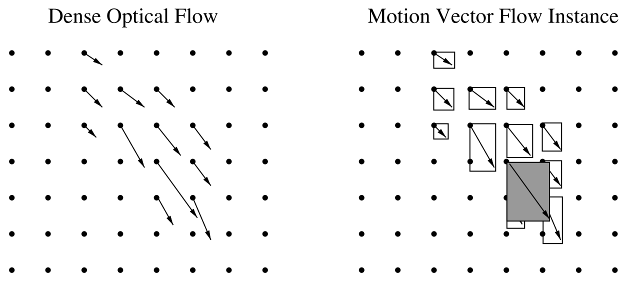
MHI (Motion History instance) templates is based upon a the algorithm described by Davis & Bobick (1997), & Bobick & Davis (2001). This template captures a time averaged cumulative history of silhouette over a determined number of image frames. We have implemented this algorithm by obtaining the silhouettes as described previously and then using a cyclic frame buffer that obtains the gradient of the silhouette within a specified interval. We found that every 5 frames was sufficient for creating effective MHI. Another critical parameter for creating MHI templates is the persistence parameter, which intuitively is memory of previous movements in the template. This obviously has a significant effect upon the performance of the template for discrimination, and we needed to fine-tune this parameter by hand in order to maximize the performance with this type of template.

While the silhouette and MHI templates have been shown to produce results for gait, they are not sensitive to either local or instantaneous velocities. Thus, we suggest a new spatio-temporal template in this paper, called MVFI (Motion Vector Flow Instance), that uses a dense optical flow in order to retain both the velocity magnitude and direction information for each frame. Since studies by Wu (2000) have shown that a fall is characterized by increases in horizontal and vertical velocities increases from two to three times daily activity, our motion template encodes velocity information and direction making it particularly sensitive to rapid body motion.

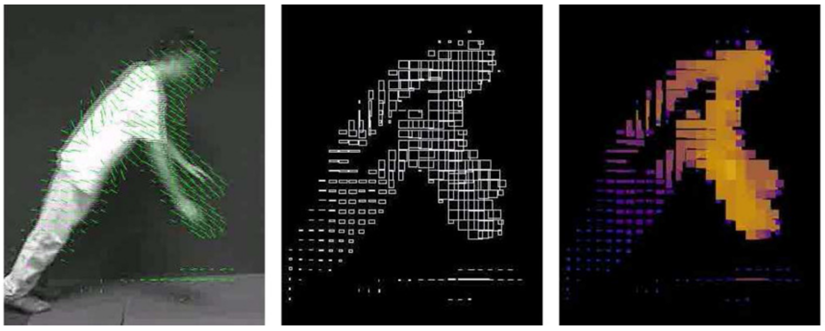
Listing 1. MVFI

```
1: InitVid $V(w,h,N_k)$;   ▷ Init video $V$, size $(w,h)$ and num. frames $N_k$
2: $z_{t=0} = O(w,h)$   ▷ Initialize output video $O(w,h)$
3: while $t < N_f$ do   ▷ Main Loop over all video frames
4:   $v_t \leftarrow V(w,h)$;   ▷ Obtain frame $v$ at $t$
5:   if $t = 0$ then $Q$ = FlowParams ()   ▷ Initialize flow model params $Q$
6:   else if $t > 0$ then
7:     $s$ = EmptyList ()   ▷ Initialize container list for MVFI boxes $s$
8:     $\mathbf{f}$=CalcOpticalFlow () $\leftarrow Q$
9:     for $x,y \in w,h$ do   ▷ Loop over flow grid $x, y$ in frame $f$
10:      $f_x, f_y = \mathbf{f}(y,x)$   ▷ Get components of flow
11:      $s \leftarrow$ ObtainMVFIboxes$(f_x,f_y)$   ▷ Push on List $s$
12:        Sort$(s, \max_{x,y}(\|\mathbf{f}\|))$   ▷ Put largest on top
13:        $z_t \leftarrow$ Write$(s)$   ▷ Write frame $z_t$ from list $s$
14:     end for
15:   end if
16:   $t \leftarrow t+1$   ▷ Get next input frame
17: end while
18: return $O(w,h)$   ▷ Return output video of MVFI
```

The essential aspects of the algorithm for constructing the MVFI template is given in Listing 1 and illustrated in Fig. 3. An input video (or video stream) $V(w,h,N_k)$, characterized by its width w, height h, and number of frames N_k, is initialized together with an output video $O(w,h)$, which shall contain the instantaneous motion template at time t, given by z_t. For each time point t, the dense optical flow field $\mathbf{f}$ is found on a grid, defined on points (x,y), as depicted in Fig. 3(a), where the motion vectors is evaluated through a well-known pyramid algorithm, given by Farnebäck (2003). Fig. 3(b)(left) shows an example video frame from a fall sequence processed with the superimposed optical flow vectors. We also



(a) Dense optical flow and MVFI vectors.



(b) Example frames of optical flow and MVFI

Fig. 3. (a) Illustration for constructing the MVFI template from dense optical flow. (b) Example frame with optical flow (left), corresponding MVFI boxes (middle), and final MVFI image frame.

re-initialize a storage list object s, that is used as a temporary container for manipulating vectors $\mathbf{f}$ at time t. For each grid point (x,y) in the optical flow, the components of the vector (f_x,f_y), together with the magnitude $\|\mathbf{f}\|$ are used in order to form a bounding box (see Fig. 3(b)(middle)) and pushed onto the list s. Subsequently, the vectors of s is sorted by the magnitude $\|\mathbf{f}\|$, pushing the largest vectors to the front of the list. Finally, the output video frame is written, encoding each flow vector in s, from smallest to largest, thereby favors large velocities, and is used to make sure that larger flow vectors will not be overwritten by raster-based writing of the final video frame. The boxes are read off the sorted list s, and the grayscale representation is written to the final output frame (see Fig. 3(b)(right)).

It should be noted that our MVFI differs from the original motion flow history (MHI) originally proposed by Bobick & Davis (2001) in two important aspects. First, we do not maintain history information through median filters, but use an instantaneous snapshot of the vectors, thereby coding rapid velocity changes more effectively. Second, the vector sizes are encoded into the size of the boxes, sorted in layers, where the largest velocities are placed on top. The pseudo-algorithm for constructing this template is given in Listing 1.

Fig. 4 shows a comparison of each of the motion templates used in this paper for a typical sequence of an exaggerated walk. As can be seen, instantaneous information of the quick motions of the arms are encoded directly with MVFI, while they can be inferred from MHI, yet not at all by the silhouette. In the following section, we shall see describe how this has a dramatic effect in the canonical eigenspace representation for discriminating high velocity motions.

3.3. Mathematics details of canonical transformation

Several detailed descriptions of the canonical eigenspace transform, based upon PCA and LDA, can be found in scientific literature (see Huang et al., 1999a) as well as recent textbooks (see Fukunaga,

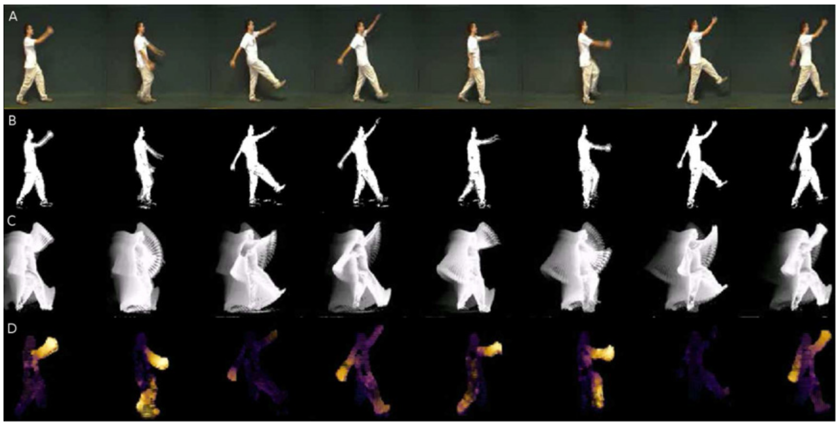


Fig. 4. Image sequence (A) for an exaggerated walk together with the different motion templates considered in this work: (B) silhouettes obtained from foreground subtraction, (C) MHI template sequence, and (D) MVFI template sequence.

1990 and Gonzalez, Woods, & Eddins (2003)). Thus, we briefly detail the mathematical framework of the technique adapted to our particular problem, yet adhering to standard notation found in these previously cited works. First, we define a motion (template) sequence, s, consisting of N_s images (or frames), where each sequence consists of a set of frames $\mathbf{x}_{i=1\cdots N_s}$. A set of sequences is depicted in Fig. 5, together with the definition of $\mathbf{x}_{i=1\cdots N_s}$. Next, we define a human action class, c, which can consists of N_s^c motion sequences. We can form a training vector as the collection of image sequences, pertaining to different classes, we can form a vector $\mathbf{X}$ consisting of the image template elements, $\mathbf{x}_{i,j}$, which is the image pertaining to the ith class, and having the jth frame within the sequence s. The total number of images in $\mathbf{X}$ is N_T, which is given by the sum $N_T = \sum_i^c N_i$. Thus, the training set, is given by the vector $\mathbf{X} = [\mathbf{x}_{1,1}\cdots\mathbf{x}_{1,N_1}, \mathbf{x}_{2,1}\cdots\mathbf{x}_{c,N_c}]$, where each $\mathbf{x}_{i,j}$ is a matrix of the pixels in the image frame (i,j). Fig. 5 illustrates the construction of the training vector $\mathbf{X}$ from the image sequences. In our implementation, the vector $\mathbf{X}$ is a column vector consisting of all the pixels from the image sequence.

The PCA canonical space is constructed from the orthogonal vectors that possess the most variance between all the images in $\mathbf{X}$. Thus, in order to construct the PCA space, we must first find the mean of the vector $\mathbf{X}$, given by $\mathbf{m_x}$, and then the covariance $\mathbf{C_x}$, representing pixels that deviate from the mean:

$$\mathbf{m_x} = \frac{1}{N_T}\sum_{i=1}^{c}\sum_{j=1}^{N_i}\mathbf{x}_{i,j}$$

$$\mathbf{C_x} = \frac{1}{N_T}\sum_{i=1}^{c}\sum_{j=1}^{N_i}(x_{ij} - m_x)(x_{ij} - m_x)^T$$

where $\mathbf{C_x}$ is then found by calculating the contribution from all pixels relative to this mean. Thus, with $\overline{\mathbf{X}} = (\mathbf{X} - \mathbf{m_x})$, we have:

$$\mathbf{C_x} = \frac{1}{N}\overline{\mathbf{X}}\overline{\mathbf{X}}^T$$

The orthogonal directions with the most variance are found from the eigen-vectors $\mathbf{u}_i$ and eigenvalues λ_i of $\mathbf{C_x}$:

$$\mathbf{C_x}\mathbf{u}_i = \lambda_i\mathbf{u}_i$$

assuming that $\mathbf{C_x}$ can be diagonalized. However, $\overline{\mathbf{X}}\overline{\mathbf{X}}^T$ is a very large $n \times n$ matrix (n is the total number of pixels of $\overline{\mathbf{X}}$). A practical and well-known simplification (see *Singular Value Decomposition* transformations from Fukunaga (1990)), is to consider a related matrix $\widetilde{\mathbf{C}}_\mathbf{x} = \overline{\mathbf{X}}^T\overline{\mathbf{X}}$, which is only $N_T \times N_T$, where we can calculate the new set of eigenvectors and eigenvalues $(\tilde{\mathbf{u}}_i, \tilde{\lambda}_i)$, that span a new vector K-dimensional space. It can be shown that this new eigenspace of $\widetilde{\mathbf{C}}_\mathbf{x}$ is in fact related to the original eigenspace of $\mathbf{C_x}$ through a simple transformation with $\mathbf{X}$.

From standard PCA theory, the K-dimensional eigenspace is truncated so that only the $k \leqslant K$ larges eigenvalues $|\lambda_1| \geqslant |\lambda_2| \geqslant \cdots|\lambda_k|$ are maintained, which is justified by demonstrating that $\lambda_j \approx 0$ for $j > k$. The partial set of eigenvectors span a space $\mathbf{y} = [\mathbf{y}_{1,1}\cdots\mathbf{y}_{c,N_c}]$, and represent projections of the original images:

$$\mathbf{y}_{i,j} = [\mathbf{u}_1\cdots\mathbf{u}_k]^T\mathbf{x}_{i,j} = \mathbf{E}\mathbf{x}_{i,j}$$

This provides a direct method for projecting the original images of the sequences onto the new multidimensional space, where each point represents an image and an image sequence represent a swarm, or trajectory.

In order to classify different types of human actions from the transformation of video sequences into points $\mathbf{y}_{i,j}$ in the new eigenspace, we apply the well-known Fisher criteria that uses the matrix $\mathbf{C_x}$ to maximize the *between class* variance ($\mathbf{S_b}$), while at the same time minimizing the *within class* variance ($\mathbf{S_w}$). It is convenient to define the ith class ϕ_i that contains all images $y_{i,j}$. Then, the two quantities are defined as follows:

$$\mathbf{S}_w = \frac{1}{N_T}\sum_{y\in\phi_i}(\mathbf{y}_{ij} - \mathbf{m}_i)(\mathbf{y}_{ij} - \mathbf{m}_i)^T$$

$$\mathbf{S}_b = \frac{1}{N_T}\sum_i N_i(\mathbf{m}_i - \mathbf{m_y})(\mathbf{m}_i - \mathbf{m_y})^T$$

In order to minimize S_w and maximize S_b simultaneously, the Fisher linear discriminant function, $\mathbf{J}(\mathbf{W})$, if given by the ratio:

$$\mathbf{J}(\mathbf{W}) = \max\frac{\mathbf{W}\mathbf{S}_b\mathbf{W}^T}{\mathbf{W}\mathbf{S}_w\mathbf{W}^T}$$

where $\mathbf{J}$ and $\mathbf{W}$ are unknowns that are found by finding the extremum: $\partial\mathbf{J}/\partial\mathbf{W} = 0$, which produces an optimal value $\mathbf{W}^*$, that satisfies this equation.

From the optimal solution, $\mathbf{W}^*$, we can write the corresponding eigenvalue equation:

$$\mathbf{S}_w^{-1}\mathbf{S}_b\mathbf{w}_i = \lambda\mathbf{w}_i$$

class separability is maximized by solving this equation. From this, we obtain once again a new orthogonal basis, spanned by $c - 1$ vectors, that takes the points $\mathbf{y}_{i,j}$ of the PCA space and transforms them to this new space, we call the LDA space, through:

$$\mathbf{z}_{i,j} = [\mathbf{v}_1, \ldots, \mathbf{v}_{c-1}]\mathbf{y}_{i,j} = \mathbf{V}\mathbf{y}_{i,j}$$

Combining the PCA and LDA transformations of the original images $\mathbf{x}_{i,j}$ from the video sequence, we can write the full canonical transformation as:

$$\mathbf{z}_{i,j} = \mathbf{VE}\mathbf{x}_{i,j}$$

Fig. 6 shows the PCA and LDA transformations of two different action sequences represented in the space of the first three larges eigenvector directions. From the example given in the figure, the

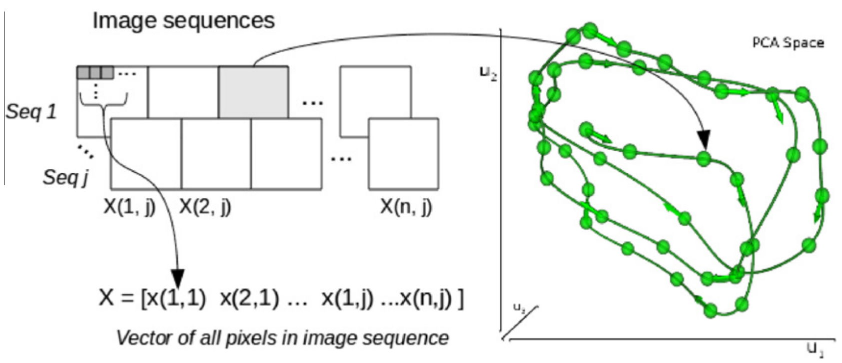
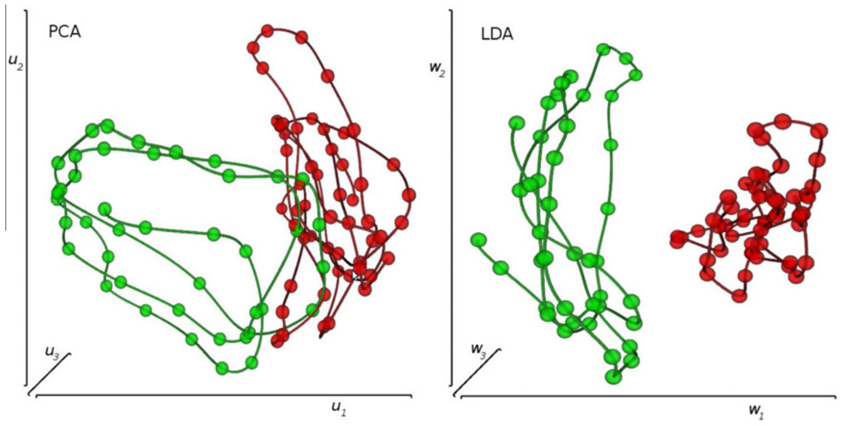


Image sequences
Seq 1
Seq j
X(1, j)
X(2, j)
X(n, j)
X = [x(1,1) x(2,1) … x(1,j) …x(n,j)]
Vector of all pixels in image sequence
PCA Space
u2
u3
u1

Fig. 5. Illustration showing the construction of $\mathbf{X}$ (all image pixels from the set of sequences), where pixels of each frame $\mathbf{x}_{i,j}$ are concatenated into $\mathbf{X}$ to form a large column vector. Consequently, each image frame is transformed to a point in the multi-dimensional canonical eigenspace.



(a) PCA transform $\mathbf{y} = \mathbf{E}\mathbf{x}$ (b) LDA transform $\mathbf{z} = \mathbf{V}\mathbf{y}$

Fig. 6. Examples of PCA and LDA transformations of two video sequences of different actions, walking (green) and jogging (red) represented in first three largest eigenvectors. The *trajectories* in the PCA overlap, while the LDA transformation separates the trajectories so as to maximize (minimize) out of (within) class variance. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

LDA transform works well at separating different classes even by using this limited space. In our implementation, we actually perform KNN based classification in a larger space (we found that $k \leqslant 10$ was sufficient).

3.4. Machine learning

As previously described, our database consisted of ≈400 video sequences, representing six different actions performed by 12 human subjects. We performed an N-fold *cross -validation* training process, where we constructed all possible binary and multiclass combinations in our dataset.

For both training and testing, it was convenient to divide the database into the following structure: *action: person: sequence: motion*, where *actions* represent the different motion classes to be trained, *person*, is a different human subject performing the action, *sequence* is the set of video sequence processed per action/person, and *motion* represent the three different spatio-temporal motion templates extracted from the video sequence. These constitute four sets:

$$\mathcal{A} = \{a_i\}_{i=1,\dots 6} \quad \text{Action set; with total number is} \quad N_A = 6$$
$$\mathcal{P} = \{p_i\}_{i=1,\dots 8} \quad \text{People set; with total} \quad N_P = 8 \text{ per action}$$
$$\mathcal{S} = \{s_i\}_{i=1,\dots 6} \quad \text{Sequence set; with total} \quad N_S = 6 \text{ per person}$$
$$\mathcal{M} = \{m_i\}_{i=1,\dots 3} \quad \text{Motion template set; with total}$$
$$N_M = 3 \text{ per sequence}$$

With this organization, we could perform multiclass training with a given combination video sequences from $\mathcal{A}, \mathcal{P}, \mathcal{S}$ and $\mathcal{M}$, while perform tests with video sequences from the complementary set. For convenience in describing our machine learning and testing method, we write the training set consists of finding the all possible combinations as the n-ary Cartesian product, or set of all ordered tuples:

$$X_1 \otimes \cdots Y_n = \{(x_1, \dots x_n, y_1, \dots y_m) | x_i \in X \text{ and } y_j \in Y\}$$

From the sets defined above, an enumeration of all possible training combinations for a given training combinations is given by:

$$\mathcal{T} = \mathcal{A} \otimes \mathcal{P} \otimes \mathcal{S} \otimes \mathcal{M}$$

where for simplicity, we shall henceforth omit the operator $\otimes$ in reference to the Cartesian product. The training software we constructed takes the values of $\mathcal{T}$, in the form $\mathcal{T} = (N(a_i), N(p_j), N(s_k), N(m_l))$ and constructs the set of all possible combinations for training. For example, a 2-class training could consist of all combinations of six different actions, but with the possibility of single or multiple people $\mathcal{P}$, and single or multiple sequences $\mathcal{S}$, which we can represent as: $\mathcal{T} = [2][6][2][1]$. We can then describe specific binary classifiers between different actions, $[a_i\ a_j]$, where for example, $[a_1\ a_6]$ is the binary classification between action a_1 and action a_6. In this case, we would then consider all combinations of $\mathcal{P} \otimes \mathcal{S} \otimes \mathcal{M}$.

The testing set is found by considering the same action set, $\mathcal{A}$, and motion template combination $\mathcal{M}$, but using the complement of the training sequences. Thus, complementary set, or testing set, $(\widetilde{\mathbf{T}})$, is found by:

$$\widetilde{T} = \mathcal{A}\widetilde{\mathcal{P}}\mathcal{S}\mathcal{M} \cup \mathcal{A}\widetilde{\mathcal{P}}\widetilde{\mathcal{S}}\mathcal{M} \quad \cup \quad \mathcal{A}\mathcal{P}\widetilde{\mathcal{S}}\mathcal{M}$$

where $\widetilde{\mathcal{P}}$, and $\widetilde{\mathcal{S}}$ are the complementary to $\mathcal{P}$ and $\mathcal{S}$ respectively.

A KNN classifier is used for determining the class of an unknown test sequence. Thus, each of the images, $\mathbf{r}_{i=1,\dots,N_v}$, of a test sequence $\mathbf{R}$, representing an unknown action, are first projected into the canonical eigenspace $\mathbf{z}(\mathcal{T})$ (of size k), constructed from a training sequence $\mathcal{T} = \mathcal{APSM}$, representing N_c distinct classes. The classification of test sequence $\mathbf{R}$ is found by the calculating the minimum Euclidean distance from the transformed test sequence $\mathbf{z}(\mathbf{R})$ to the trained class trajectories $\mathbf{z}(\mathcal{T})$.

4. Experimental results

We performed multiclass training between all possible combinations of human actions in this study, for N_A = 2, 3, 4, 5, 6. Moreover, for each multiclass training combination and motion template, we performed an N- fold cross validation between all video sequences in our dataset obtained from different people as described in the previous section. From these extensive tests, we could understand how average recognition results are effected by adding sequences from different people to the training. We were also interested in understanding how different clothing, body shapes, and lighting could effect the final results. By training and and testing with all possible combinations, we were able to observed trends in the results, which would have been impossible otherwise. In this sections, we show the most important recognition results between the different motion templates in our study.

In order to appreciate the size of the typical training sets quoted in this section, Table 1 shows an execution summary for a few combinations in the training process. The first column, is the combination $\mathcal{T} = (N(a_i), N(p_j), N(s_k), N(m_l))$. For example, $\mathcal{T} = (2, 6, 1, 1)$ represents all combinations of binary action classification, using six people, one sequence for each person, and for each motion template. The second column of Table 1 is the total number of

Table 1
Summary of some representative training statistics, showing the average number of images, the average training times, and total run times.

$\mathcal{T}$	Number comb.	Avg. total images	Avg. train time (s)	Total train time (s)
2, 1, 1, 1	360	51.3	2.1	790.2
2, 6, 1, 1	1723	196.9	19.1	33043.5
3, 1, 1, 1	480	77.0	3.7	1795.8
3, 6, 1, 1	1319	400.2	63.6	84006.3
5, 1, 1, 1	288	128.3	7.6	2201.6
5, 6, 1, 1	339	659.2	139.5	47318.4

combinations (obtained with the Cartesian product), so that the example (2,6,1,1) has 1723 different combinations. The third through fifth columns of Table 1 provide average number of images for each case (since the number of frames in each sequence is typically different), average training time, and the total training time, respectively. All the training runs were carried out on a typical PC (Intel Quad3.2 GHz).

As can be seen, the average number of images for binary training ($N_A = 2$) is from ~50 images to ~200 images (lower bound), with average training times from $t_{train} \sim 2$ s to $t_{train} \gtrsim 20$ s. For multiclass training, however ($N_A = 3,4,5,6$), the number of images in the training vector is larger. For example for the case of $N_A = 5$, the average number of images for (5,1,1,1) is ~130, while the average number for (5,6,1,1) is $\gtrsim 650$ images, which increases the computational costs. In these cases, and with the large number of possible combinations (as in $N_A = 3$), the training process took several days of continual execution.

The execution time for a test sequence is much smaller than the training values shown in Table 1, since a typical image sequence contains between ~50 to ~70 frames. Thus, the average execution time for tests is $t_{test} \sim 0.25$ s, making the eigenspace method plausible for real time detection applications.

4.1. Multiclass comparisons

Multiclass recognition results are obtained by running the corresponding complementary tests $\widetilde{\mathcal{T}}$ for all training combinations $\mathcal{T}$, as described in the previous section. For an individual test sequence, the minimum euclidean distance is used as a metric for determining class membership. Thus, in order to determine if a sequence s pertains to a particular class, all the points in its *trajectory* in the space $\mathbf{z}$ are considered. In our system, s belongs to class C if at least 50% of the points in the trajectory $\mathbf{z}(s)$ are closest to C, in which case the entire sequence is classified with certainty. Thus, the recognition rates quoted in this section refer to the fraction of the ensemble of test sequences that are correctly classified.

Fig. 7 shows a comparison of representative (*normalized*) histograms of all the possible three class combinations (i.e. $N_A = 3$) as a function of the recognition rates obtained from the corresponding complementary tests for the three different motion templates (Silhouette, MHI, and MVFI). Similar histograms are obtained for all the possible combinations $\mathcal{T}$ in the database. From each histogram, the *sampled* mean and standard deviations are extracted for later comparison.

The examples histograms of Fig. 7 demonstrate a few general performance features of the different motion templates. First, the recognition rates are highest between the silhouette templates and the MVFI templates. Second, and more importantly, the standard deviation is smallest for our proposed MVFI template method, peaking around 100% recognition.

4.2. Motion template comparisons

Since there is a large number of combinations for multiclass training/testing, different aspects of our experimental data can be studied against various parameters, such as specific combinations of people and sequences N_p or between specific action classes. As an example, we first obtain a comparison of the recognition rates from different action classes N_A as a function of the number of unique persons N_P included in the training for each motion template. By using the extracted means of the normalized histograms, Fig. 8 shows a comparison of the recognition rates with different motion templates as the number of image sequences from different people N_p in the training process is increased for different multiclass cases (varying N_A). In all cases the MVFI template outperforms the other templates. Of particular interest is the case for $N_A = 5$ (training 5 action classes simultaneously), the MVFI performs considerably better than the other motion templates.

In order to determine how well each motion template performs for multiclass recognition, Fig. 9 shows direct comparison for each template as a function of the number of sequences of different persons N_p included in the training. Once again, the MVFI template is superior to the other templates for $N_A = 2$ through $N_A = 5$.

4.3. Fall detection for binary classification

Binary classification between falls and other actions in the study provides a direct method for understanding how the velocity information encoded in the MVFI template is successful for discriminating actions with large motion changes. Fig. 10 shows comparisons of the motion templates for different binary classifications of actions a_i versus the fall action. The recognition rate is the fraction of correctly classified test sequences from either the action a_i or the fall action. As in the previous plots, the recognition rates are obtained from the means and variance obtained from normalized histograms of tests sequences not included in the training sets.

As can be seen from Fig. 10, the MVFI motion template outperforms all other motion templates for detecting falls. For cases where the mean values are close to the values obtained with sil-

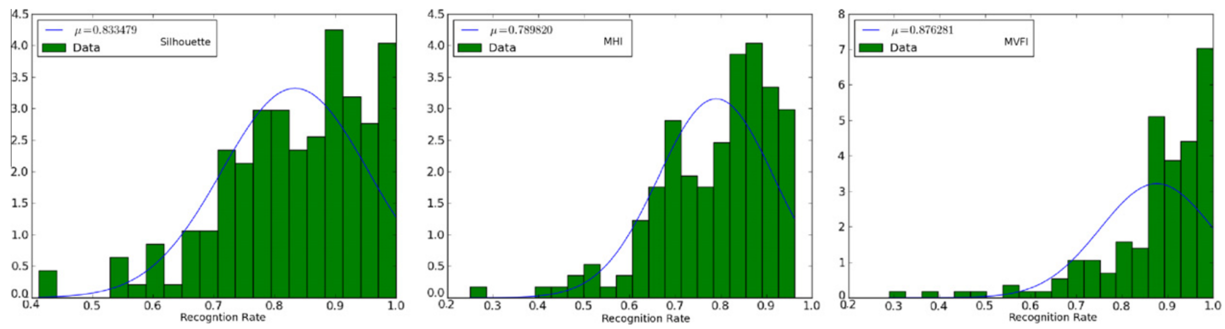


Fig. 7. Normalized histograms of recognition rates obtained from the three different spatio-temporal motion templates used in the study: silhouette (left), MHI (middle), and MVFI (right), from all combinations of 3 action classes, 1 person, and 1 sequence ($\mathcal{T} = (3,1,1,1)$).

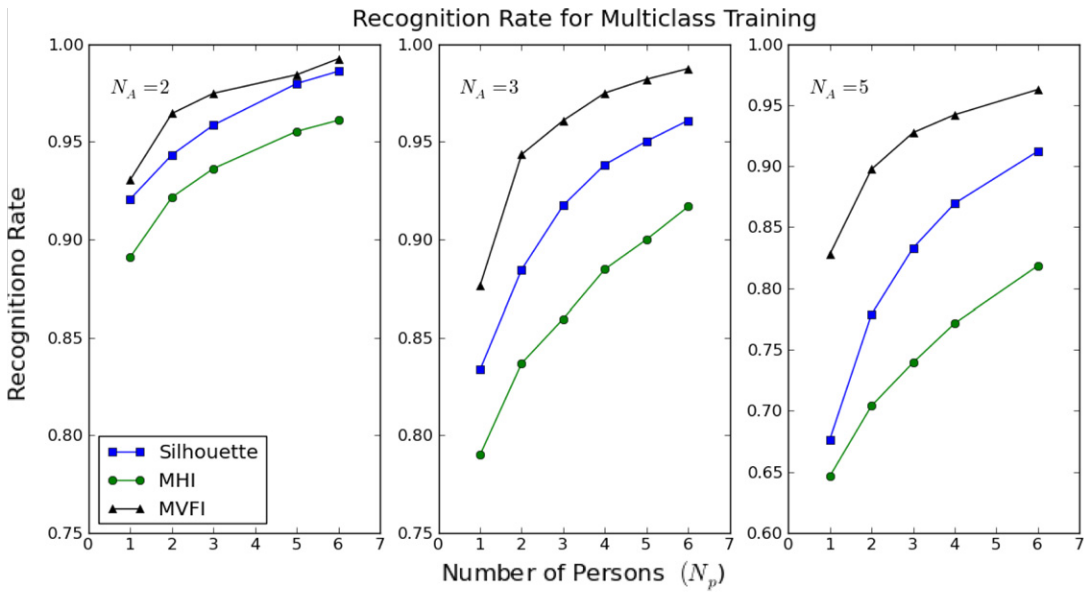


Fig. 8. Comparison of recognition rates for different multiclass training: number of actions N_A as a function of different number of persons N_P included in training.

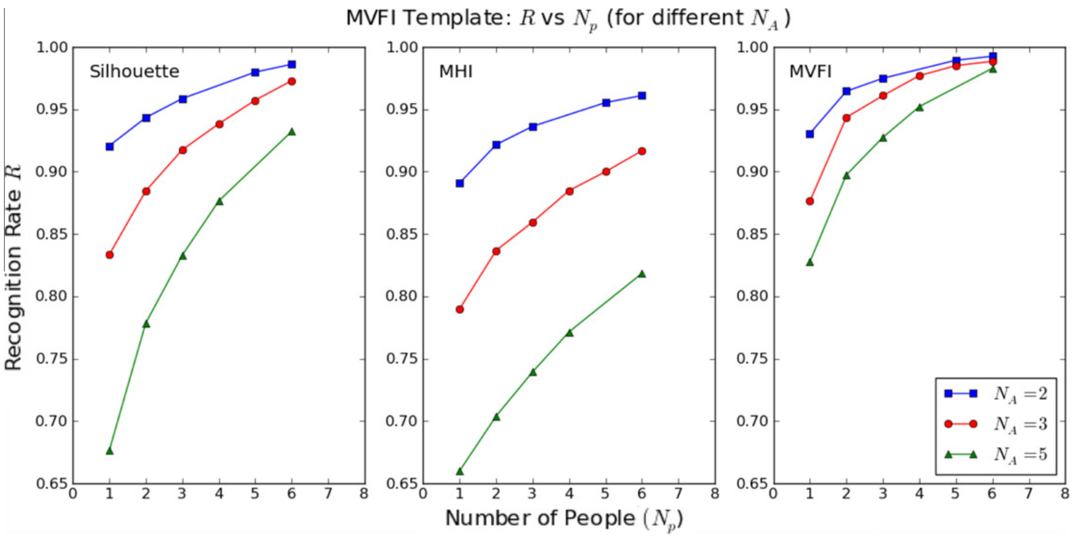


Fig. 9. Comparison of recognition rates for different the different motion templates (silhouette, MHI, MVFI) as a function of incrementally including more people in the training set.

houette templates, the standard deviations are smaller. In particular, we see that the motions that are similar in appearance, such as lying down and bending over, the MVFI is the template of choice since it achieves near 100% recognition rates since it is sensitive to velocity changes. A notable comparison is the case of lying down as compared with falls. In this case, the silhouette achieves ~92% recognition rate, but with a large standard deviation, while the MVFI is approximately 100% accurate with very small standard deviation.

4.4. Multiclass detection

If the detection of falls is to have practical use, it must be able to distinguish multiple classes with high precision. For illustrative purposes, Fig. 11 shows the PCA and LDA spaces for a 6-class training case, where one sequence per action is obtained for one person. It is notable in the PCA space, there are several classes that overlap,

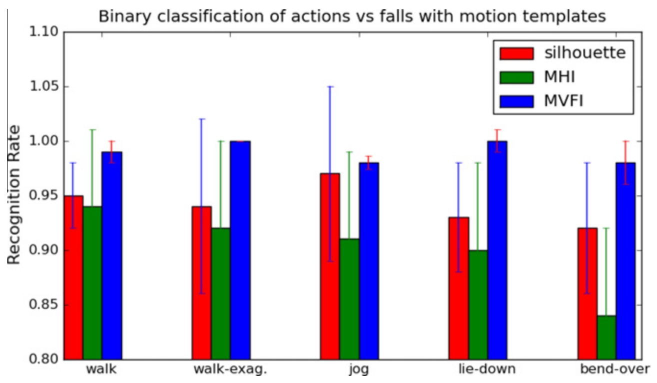


Fig. 10. Comparison of motion templates for binary classification of actions.

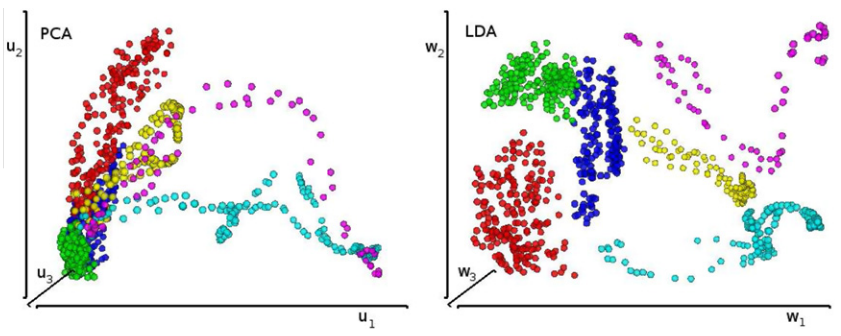


Fig. 11. The PCA and LDA training space with six different action classes of this study.

Table 2
Comparison of the average recognition rates (in %) from the different motion templates using a 10-fold cross validation with multiclass training ($N_A = 6$).

Template	Actions					
	Walk (%)	Walk (2) (%)	Lying (%)	Fall (%)	Sit (%)	Bend (%)
Silhouette	90.9	93.6	92.1	92.0	89.1	88.7
MHI	87.4	88.5	81.9	85	85.5	81.2
MVFI	99.2	99.2	97.8	99	96.4	98.1

while the LDA transformation separates these classes considerably. In a real case, there would be many more trajectories included in the training that include different persons and thus a larger range of variability in the classes.

We performed a 10-fold cross validation for all six action classes and all the training sequences of our database. Table 2 shows the performance of the different algorithms. As can be seen, the MVFI outperforms all other motion templates.

5. Conclusions and further research

For the cases we considered, our recognition rates are consistent with those from similar work, as recently reported in Ahmad & Lee (2010). Nonetheless, a direct comparison would require comparison of the reported methods on our datasets as well as the same method for quoting the recognition rates as in this paper to make the comparison meaningful.

This paper has compared two different motion templates described previously with a new spatio-temporal motion template, MVFI, which we have proposed as an improvement over previous techniques. We have demonstrated that the MVFI outperforms other methods for detecting actions characterized by large velocities. Thus, this work suggests that it is important to preserve velocity information in each image sequence if we wish to use eigenspace techniques for discriminating human motion.

While our database consists of several different people, with different clothing and body types, we have shown that the MVFI motion template works well in all situations for action recognition. A shortcoming of the database is the fact that all actions were taken with the same camera distance to the subject. Thus, future studies shall consider both different camera angles and distances.

References

Ahad, M. A. R., Tan, J. K., Kim, H. S., & Ishikawa, S. (2009). Temporal motion recognition and segmentation approach. *International Journal on Imaging Systems and Technologies, 19*(2), 91–99.

Ahmad, M., & Lee, S. W. (2010). Variable silhouette energy image representations for recognizing human actions. *Image and Computer Vision, 28*(5), 814–824.

Bobick, A.F., & Davis, J.W. (1996). An appearance-based representation of action. In *Proceedings of the 13th international conference on pattern recognition (ICPR)* (pp. 307–312).

Bobick, A. F., & Davis, J. W. (2001). The recognition of human movement using temporal templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 23*(3), 257–267.

Bourke, A., O'Donovan, K., & OLaighin, G. (2007). Distinguishing falls from normal ADL using vertical velocity profiles. In *29th Annual international conference of the ieee engineering in medicine and biology society (EMBS)* (pp. 3176–3179).

Bradski, G. (2000). The OpenCV library. Dr. Dobb's Journal of Software Tools.

Chen, D., Bharucha, A.J., & Wactlar, H.D. (2007). Intelligent video monitoring to improve safety of older persons. In *Proceedings of the 29th Annual international conference of the IEEE engineering in medicine and biology society (EMBS)* (pp. 3814–3817).

Cho, C. W., Chao, W. H., Lin, S. H., & Chen, Y. Y. (2009). A vision-based analysis system for gait recognition in patients with parkinson's disease. *Expert Systems with Applications, 36*(3), 7033–7039.

Das, S. R., Wilson, R. C., Lazarewicz, M. T., & Finkel, L. H. (2006). Two-stage PCA extracts spatiotemporal features for gait recognition. *Journal of Multimedia, 1*(5), 9–17.

Davis, J., & Bobick, A. (1997). The representation and recognition of human movement using temporal templates. In *Proceedings of the 1997 IEEE computer society conference on computer vision and pattern recognition* (pp. 928–934).

Deutscher, J., & Reid, I. (2005). Articulated body motion capture by stochastic search. *International Journal of Computer Vision, 61*(2), 185–205.

Etemad, K., & Chellappa, R. (1997). Discriminant analysis for recognition of human face images. In *Audio and video-based biometric person authentication. Lecture Notes in Computer Science* (Vol. 14 (8), pp. 1724–1733).

Farnebäck, G. (2003). Two-frame motion estimation based on polynomial expansion. In *Proceedings of the 13th Scandinavian conference on image analysis* (pp. 363–370).

Felzenszwalb, P., Girshick, R., McAllester, D., & Ramanan, D. (2010). Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 32*(9), 1627–1645.

Forsyth, D., Arikan, O., Ikemoto, L., O'Brien, J., & Ramanan, D. (2006). Computational studies of human motion: Part 1, tracking and motion synthesis. *Foundations and Trends in Computer Graphics and Vision, 1*(2/3), 77–254.

Fukunaga, K. (1990). *Introduction to statistical pattern recognition* (2nd ed.). San Diego, CA, USA: Academic Press.

Gonzalez, R. C., Woods, R. E., & Eddins, S. L. (2003). *Digital Image Processing Using MATLAB*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc.

Han, L., Wu, X., Liang, W., Hou, G., & Jia, Y. (2010). Discriminative human action recognition in the learned hierarchical manifold space. *Image and Vision Computing, 28*(5), 836–849.

Huang, P.S., Harris, C.J., & Nixon, M.S. (1999a). Human gait recognition in canonical space using temporal templates. In *IEE Proceedings of vision, image and signal processing* (Vol. 146 (2), pp. 93–100).

Huang, P. S., Harris, C. J., & Nixon, M. S. (1999b). Recognising humans by gait via parametric canonical space. *Artificial Intelligence in Engineering, 13*(4), 359–366.

İkizler, N., & Forsyth, D. (2008). Searching for complex human activities with no visual examples. *International Journal of Computer Vision, 80*(3), 337–357.

Kaewtrakulpong, P., & Bowden, R. (2001). An improved adaptive background mixture model for real-time tracking with shadow detection. In *Second European workshop on advanced video based surveillance systems (AVBS)* (pp. 149–158).

Kellokumpu, V., Zhao, G., & Pietikäinen, M. (2010). Dynamic textures for human movement recognition. In *Proceedings of the ACM international conference on image and video retrieval (CIVR)* (pp. 470–476).

Larson, L., & Bergmann, T. F. (2008). Taking on the fall: The etiology and prevention of falls in the elderly. *Clinical Chiropractic, 11*(3), 148–154.

Leone, A., & Distante, C. (2007). Shadow detection for moving objects based on texture analysis. *Pattern Recognition, 40*(4), 1222–1233.

Li, L., Huang, W., Gu, I.Y.H., & Tian, Q. (2003). Foreground object detection from videos containing complex background. In *Proceedings of the eleventh ACM international conference on multimedia* (pp. 2–10).

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

Meeds, E., Ross, D., Zemel, R., & Roweis, S. (2008). Learning stick-figure models using nonparametric bayesian priors over trees. In *Proceedings of the EEE conference on computer vision and pattern recognition (CVPR)* (pp. 1 –8).

Moylan, K. C., & Binder, E. F. (2007). Falls in older adults: Risk assessment, management and prevention. *The American Journal of Medicine, 120*(6), 493–497.

Murase, H., & Sakai, R. (1996). Moving object recognition in eigenspace representation: Gait analysis and lip reading. *Pattern Recognition Letters, 17*(2), 155–162.

Noury, N., Fleury, A., Rumeau, P., Bourke, A., Laighin, G., Rialle, V., & Lundy, J. (2007). Fall detection - principles and methods. In *29th Annual international conference of the IEEE engineering in medicine and biology society (EMBS)* (pp. 1663 –1666).

Nyan, M., Tay, F., Tan, A., & Seah, K. (2006). Distinguishing fall activities from normal activities by angular rate characteristics and high-speed camera characterization. *Medical Engineering and Physics, 28*(8), 842–849.

Poppe, R. (2010). A survey on vision-based human action recognition. *Image and Vision Computing, 28*(6), 976–990.

Venkatesh Babu, R., & Ramakrishnan, K. R. (2004). Recognition of human actions using motion history information extracted from the compressed video. *Image and Vision Computing, 22*(8), 597–607.

Williams, A., Ganesan, D., & Hanson, A. (2007). Aging in place: Fall detection and localization in a distributed smart camera network. In *Proceedings of the 15th international conference on multimedia* (pp. 892–901).

Wu, G. (2000). Distinguishing fall activities from normal activities by velocity characteristics. *Journal of Biomechanics, 33*(11), 1497–1500.