

# Human action recognition by means of subtensor projections and dense trajectories

Josep Maria Carmona<sup>a,\*</sup>, Joan Climent<sup>a</sup>

<sup>a</sup>*BarcelonaTech, c/ Llorens i Artigas, Barcelona 08028, Spain*

---

## Abstract

In last years, most human action recognition works have used dense trajectories features, to achieve state-of-the-art results. Histograms of Oriented Gradients (HOG), Histogram of Optical Flow (HOF) and Motion Boundary Histograms (MBH) features are extracted from regions and being tracked across the frames.

The goal of this paper is to improve the performance obtained by means of Improved Dense Trajectories (IDTs), adding new features based on temporal templates. We construct these templates considering a video sequence as a third-order tensor and computing three different projections. We use several functions for projecting the fibers from the video sequences, and combined them by means of sum pooling.

As a first contribution of our work, we present in detail the method based on tensor projections. First, we have assessed the results obtained using only template based action recognition. Next, in order to achieve state-of-art recognition rates, we have fused our features with those of IDTs. This is the second contribution of the article.

Experiments on four different public datasets have shown that this technique improves IDTs performance and that the results outperform the ones obtained by most of the state-of-the-art techniques for action recognition.

### *Keywords:*

action recognition, subtensors, dense trajectories, keypoint descriptors, temporal template

---

\*Corresponding author

*Email address:* josep.maria.carmona@estudiant.upc.edu (Josep Maria Carmona)

## 1. Introduction

Visual analysis of human movements concerns the understanding of human activities from image sequences. The automatic classification and localization of human actions or gestures is useful for multiple applications such as video surveillance, human-computer interaction, video indexing, browsing gait, or analysis for biometrics. Many approaches compute local features extractors to represent the visual pattern of different regions of the frames of an image sequence [1, 2].

Wang et al. presented in [3] *dense point trajectories*, where features are extracted from regions which are being tracked using optical flow across the frames. They extracted static and motion information combining trajectory, Histogram of Oriented Gradients (HOG), Histogram of Optical Flow (HOF), and Motion Boundary Histograms (MBH) feature descriptors. The same authors presented in [4] an improvement of their previous approach, using camera motion to correct the trajectories. This technique, called *improved dense trajectories* (IDTs), provides the state-of-the-art of some public datasets and it has been used for many authors in their approaches, directly or indirectly.

In this paper we present a template based approach for the recognition of human actions that, combined with IDTs, improves the state-of-the-art results in action recognition. The templates are computed from three different projections considering a video sequence as a third-order tensor. We compute each projection from the fibers of the tensor using a combination of simple functions. Fibers are subtensors formed by fixing every index but one. With such projections we obtain a 3-component template. The main goal of this paper is to combine the descriptors taken from our templates with HOG, HOF and MBH in order to improve the results of IDTs. First, we have studied our technique based on temporal templates in isolation, in order to optimize its performance, and next, we have added it to IDTs and assessed the performance of the complete technique. As a first step, we have tested five different simple functions used to project the fibers, namely, *supremum*, *mean*, *standard deviation*, *skewness* and *kurtosis* (i.e., function maximum and moments from 1 to 4). We have studied the significance of the templates obtained using the five functions, assessing the performance improvement obtained when we combine them using sum pooling.

Once the projections have been computed, we obtain a robust representation of human actions using feature descriptors of the keypoints extracted

from the template triplets obtained. The choice of which feature descriptor is the most suitable for our application, is one of the objectives of our work. We have studied the performance obtained using four feature descriptor techniques (PHOW, LIOP, HOG and SMFs) on the triplets.

With the descriptors obtained from the templates, we feed a linear Support Vector Machine (SVM) classifier using Fisher Vectors encoding ( $Fv$ ) [5].  $Fv$  encoding has proven to give more accurate results than a Bag of Features (BoF) approach. The features obtained using our method based on templates can easily be combined with HOG, HOF or MBH, and added to IDTs to improve its performance. Therefore, we have finally combined our descriptors with IDTs encoded with  $Fv$ , and compared our results against other successful action recognition techniques.

Figure 1 shows a block diagram of the whole method. As we can see, first, three projections are computed from the input video sequence applying different simple functions. Each simple function projects a stream of data. For each stream, local feature descriptors are computed. These features are later clustered using Gaussian Mixture Model (GMM) [6], and encoded using Fisher Vectors ( $Fv$ ). The normalized  $Fv$  of each stream are fused using sum pooling and entered into a SVM. Finally, the fused features are combined with the IDTs (HOG, HOF and MBH) features summing both SVM scores.

The main contributions of this work are detailed below:

As a first contribution, we consider a video sequence as a third-order tensor and we compute three templates from the fibers of this tensor using simple functions. We propose this template based approach for the recognition of human actions. We have studied the performance of our templates using five different functions (*supremum*, *mean*, *standard deviation*, *skewness* and *kurtosis*). To do this, we have assessed the most suitable tract width and the most suitable feature detector/descriptor. We have assessed the most suitable feature representation ( $Fv$  or BoW), and we have also studied two different scenarios, namely, *Mfunction* and *Sfunction*.

As a second contribution, we have combined our features with the ones used by IDTs (HOG, HOF and MBH), and we have proved the complementarity of both.

The article is organized as follows: in next section, we look over the literature related with the presented work. The description of the proposed method is detailed in Section 3. Section 4 shows the experimental setup giving the necessary details to reproduce the results presented in section 5. The experimental results state that the technique presented outperforms

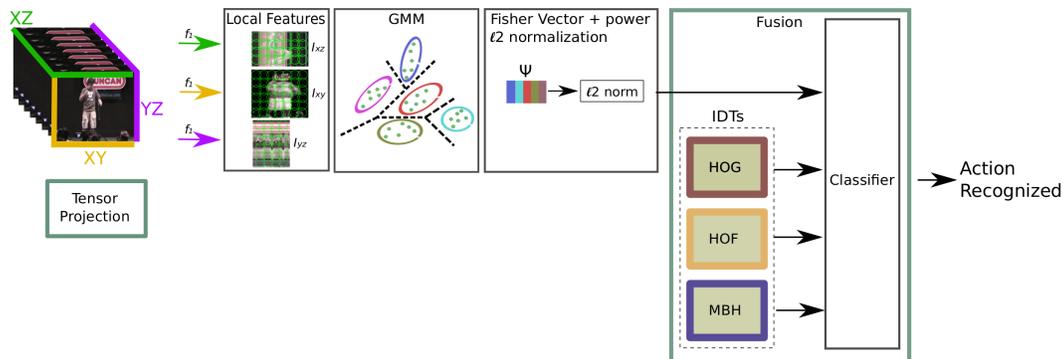


Figure 1: Block diagram of OUR approach.

most of the state-of-art methods in action recognition. Finally, section 6 concludes this paper.

## 2. Related work

Some authors have tried to collapse the temporal motion information of a whole sequence into a single image template. Techniques based on templates convert the video sequence into a static shape pattern. These techniques are easy to implement and require low computational load.

Originally, Bobick & Davies introduced temporal templates in [7]. They consist of 2D images computed from 3D image sequences, retaining important temporal information. In their work, they presented two temporal templates called Motion Energy Images (MEI) and Motion History Images (MHI). MEI is a binary image encoding the areas where motion occurred, and MHI is a grey-level image encoding how recently motion occurred at a pixel. In the same vein, Fernando et al. captured the temporal ordering of actions by training a linear ranking machine in [8]. A drawback of MHI is that it only encodes the time from the last observed motion at every pixel. To address this limitation, [9] introduced Motion History Histograms (MHH), which stores frequency information as the number of times motion is detected at every pixel, further categorized into the length of each motion. [10] proposed a weighted MHI/MEI that use fuzzy functions to emphasize motion information in various temporal regions instead of the last frames as traditional MHI. Recently, Bilen et al.[11] proposed a long-term pooling in a neural network context. They condense a video sequence in a single image, named

Dynamic Image, obtained as a ranking classification that sorts video frames temporally.

One way to summarize the information of a video sequence is temporal pooling [7, 11, 12, 13, 8]. Temporal pooling consists in computing a function, such as *supremum* or *mean*, on the features over a temporal segment of a video sequence. In [12], authors used the distribution of classifier scores instead of *max* or *mean* functions. They ordered and combined these ones using a weight vector learned from the training dataset. Ryoo et al.[13] proposed a new feature representation named Pooled Time series (PoT) that given a sequence of per-frame descriptors from a video, abstracts it computing changes in each descriptor.

There are also other authors that have used specific techniques characterizing an action sequence as a third order data tensor. In [14], they investigated video volume matching, extending Canonical Correlation Analysis (CCA) to the tensor framework, by developing a Tensor Canonical Correlation Algorithm (TCCA). Since, CCA cannot consider the nonlinear correlation between multiview features and with the purpose of reducing dimensionality of extracted visual features, [15] proposed Hessian Multiset Canonical Correlations (HesMCC) for multiview dimension reduction. HesMCC can exploit the intrinsic local geometry of the data manifold in contrast to Laplacian. In [16], they developed a General Tensor Discriminant Analysis (GTDA) as a preprocessing step for Linear Discriminant Analysis (LDA) for gait recognition. To represent gaits, they developed three different Gabor-function-based image representations, namely, GaborD to respond about direction information, GaborS to respond about scale information and GaborSD to respond about both. The characterization of the underlying geometry of the tensor space from raw pixels was presented in [17] with successful results. They abstracted an N order tensor as a point on a product manifold where the number of factors is given by the order of the tensor. Action classification was then performed on the basis of geodesic distance on the product manifold associated with an action video.

An important reason for the success of object recognition techniques [18] has been the robustness of the feature extractors and descriptors, like the Scale Invariant Feature Transform (SIFT) algorithm [19], Histogram Of Oriented Gradients (HOG) [20, 21], Local Intensity Order Pattern (LIOP) [22], Local Binary Patterns [23] or some variants of the previous techniques, like Pyramid Histogram Of visual Words (PHOW) [24]. The success of these techniques motivated the use of them for action/gesture recognition, extend-

ing these techniques to video sequences. In [25] they used 3D SIFT for action recognition using spatio-temporal features. In [26], they extended LBP to Volume Local Binary Patterns (VLBP), combining motion and appearance, but in this case they applied them to recognize dynamic textures. To make the VLBP computationally simple and easy to extend, only the co-occurrences on three separated planes were considered. The textures were modelled with concatenated Local Binary Pattern histograms from Three Orthogonal Planes (LBP-TOP). SIFT, HOG and PHOW were combined with Bag of Words (BoW) [27] paradigm and applied to action recognition successfully in [28].

In [3], they proposed to sample feature points on a dense grid in each frame and track them using optical flow algorithm to improve other tracking techniques as KLT tracker. They densely sampled feature points on different scales and then these feature points were tracked on each scale separately. Points of every frame were concatenated to form trajectories (DTs). They combined different features like trajectory shape, HOG, HOF and MBH [29] within space-time volumes aligned with these trajectories using spatio-temporal pyramids. HOG captured the static appearance, HOF represented motion, and MBH represented motion boundary information. MBH splits the horizontal and vertical OF components, spatial derivatives of the both components were computed and orientation information was quantized into two histograms. The magnitude of the OF was used for weighting.

In [4], they presented improved dense trajectories (IDTs) for action recognition. To estimate camera motion they matched feature points using SURF descriptor and dense optical flow, and then they improved the estimation using a human shape detector. IDTs improve the performance with respect DTs considering camera motion correction.

Peng et al.[27] assessed the fusion of HOG, HOF and MBH in three different levels named, descriptor level, representation level, and score level. Descriptor level is performed in the cuboid levels, concatenating multiple descriptors from a cuboid to a single descriptor. In representation level, the fusion is performed in the video level entering the descriptors into BoF separately, and fusing the global representation as a single descriptor. For score level, fusion is performed in the video level also but each descriptor trains a classifier and the final classification is obtained fusing the scores of multiple classifiers. They also assessed different feature encoding like Vector Quantization encoding (VQ) [30], Localized Soft Assignment Encoding (SA-k) [31], Locality-constrained Linear Encoding (LLC) [32], Fisher Vec-

tor ( $Fv$ ) [33], Vector of Locally Aggregated Descriptors (VLAD) [34] and Super-Vector Coding (SVC) [35] and obtained the best results using  $Fv$  encoding and representation level fusion, therefore we have used this one in our implementation of IDTs.

Peng et al.[36] extracted the improved dense trajectory features of [4] and encoded them using  $Fv$  and VLAD separately. Finally they concatenated  $Fv$  and VLAD into a Hybrid Super Vector. Wu et al. [37] evaluated various improvement techniques for VLAD based video encoding and suggested use VLAD or  $Fv$  instead of BoW. Peng et al. had already proposed in [27] a simple feature representation called hybrid representation, exploring the complementarity of different Bag of Visual Words models frameworks and local descriptors.

Hyun-Joo et al.[38] proposed a new model named Bag-of-Sequencelets (BoS). A BoS model represents a video as a sequence of Primitive Actions (PA), considering a complex action like a temporally ordered composition of sub-actions. To enable PA learning, they represent a video sequence as a sequence of IDTs features and Fisher encoding representation. In [39], they propose a generalized version of Laplacian regularized sparse coding for human activity recognition called p-Laplacian regularized sparse coding (pLSC). They demonstrated that the proposed pLSC outperforms the conventional Laplacian regularized sparse coding and Hessian regularized sparse coding algorithms.

Another group of techniques is inspired by biological processes. The input images are convoluted, and then a pooling operation is applied to the outputs [40, 41, 42, 11]. Authors proposed in [40] a bioinspired technique called standard model features (SMFs), that attempts to summarize a core of well-accepted facts about the ventral stream in the visual cortex. Even though this technique has also been extended to video sequences achieving successful results [43], the main drawback is its large computation time.

Convolutional Neural Networks (CNNs) are other bioinspired techniques that have been also extended to action recognition [42, 11, 41]. Ng et al.[41] proposed feature-pooling and recurrent neural networks to combine image information to handling full length videos. They explored also the necessity of motion information and they confirmed that for UCF101 dataset is necessary to use optical flow to achieve acceptable results. In their approach they combined spatial and temporal information. Moreover, they showed that Long Short-Term Memory (LSTM) over a temporal features pooling doesn't improve (or improves only marginally) the results depending on the dataset

used.

Simonyan et al. [42] also combined spatial and temporal information using a new architecture based on two separate recognition streams (spatial and temporal) combined by late fusion. They state that using optical flow as input, their model does not require significant hand-crafting, but they say also that essential ingredients of shallow representation like local feature pooling over spatio-temporal tubes, are missed in their architecture.

Zhichen et al.[44] proposed a semantic learning algorithm for action recognition from still images. They defined "semantic part" as any region that provides great contribution to the right recognition. They used CNNs pooling to find interactive objects and poses information and then they combined both to form a more discriminative representation.

In [11], authors presented a new video representation named dynamic images that condense the video sequences in a single image using rank pooling and CNNs. They proposed also a temporal pooling layer extending dynamic images to the CNN feature maps.

Obtaining information from multiple views computed from independent data sources or from the same input data, is a major difference between multi-view and single-view learning algorithms. The complementary principle of multi-view setting states that each view of the data may contain some knowledge that other views do not have. Therefore, multiple views can be employed to comprehensively and accurately describe the data [45, 46].

In [46], they presented a method that detects general and fine-grained human action recognition in video sequences. First, they estimate human pose and human parts positions in video sequences and crops different scaled patches. They used six patches (three from RGB and three from OF) for obtaining appearance and motion information. Then these six patches fed a CNN to process each image patch.

### 3. Our approach

#### 3.1. Tensor projections

A video sequence can be naturally represented as a third order tensor  $\mathbf{X} \in I \times J \times K$  where  $I$ ,  $J$  and  $K$  are the image width, image height, and sequence length, respectively. We will use  $a_{ij}$  to represent the element  $(i, j)$  of a matrix  $\mathbf{A}$  and  $\mathbf{x}_{ijk}$  to represent the element  $(i, j, k)$  of a third-order tensor  $\mathbf{X}$ .  $\mathbf{A}^{(n)}$  represents the  $n$ th matrix of a third-order tensor. Subtensors are formed when a subset of the indices are fixed. A third-order tensor can

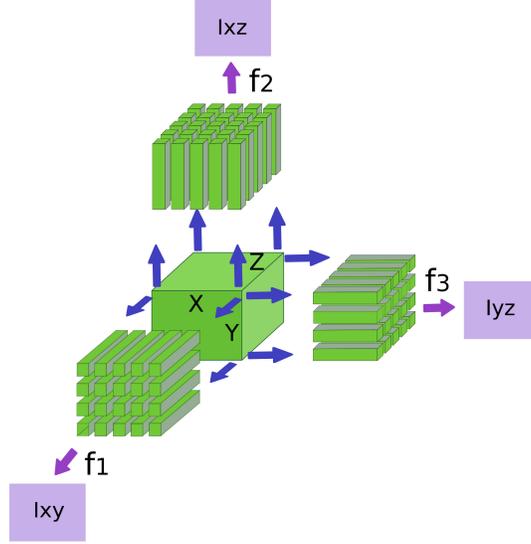


Figure 2: Sequence projected into 3 single views.

be considered as a set of vectors called fibers. A fiber is defined by fixing every index except one. Third-order tensor has column, row and depth fibers denoted by  $\mathbf{x}_{:jk}$ ,  $\mathbf{x}_{i:k}$  and  $\mathbf{x}_{ij:}$  respectively, where colon indicates all elements of a mode. Therefore, the mode1 is the subspace spanned by all height fibers, the mode2 is the subspace spanned by all width fibers and mode3 is the subspace spanned by all depth fibers. Figure 2 shows the fibers of a third-order tensor.

Lets define a tract as a set of neighbouring fibers. Equations 1, 2 and 3 show the tracts of mode3, mode2 and mode1 denoted by  $T_{xy}$ ,  $T_{xz}$  and  $T_{yz}$  respectively.

$$T_{xy}^{\beta}(i, j) = \bigcup_{r=i-\frac{\beta}{2}}^{i+\frac{\beta}{2}} \bigcup_{s=j-\frac{\beta}{2}}^{j+\frac{\beta}{2}} (x_{r:s}) \quad (1)$$

$$T_{xz}^{\beta}(i, j) = \bigcup_{r=i-\frac{\beta}{2}}^{i+\frac{\beta}{2}} \bigcup_{s=j-\frac{\beta}{2}}^{j+\frac{\beta}{2}} (x_{r:s}) \quad (2)$$

$$T_{yz}^\beta(i, j) = \bigcup_{r=i-\frac{\beta}{2}}^{i+\frac{\beta}{2}} \bigcup_{s=j-\frac{\beta}{2}}^{j+\frac{\beta}{2}} (x_{:rs}) \quad (3)$$

With  $\beta$  being the neighbourhood width of fibers that form the tract. Equations 4, 5, and 6 are used to compute the projection triplet, where  $f_1 = \mathbb{R}^{\beta \times \beta \times K} \rightarrow \mathbb{R}$ ,  $f_2 = \mathbb{R}^{\beta \times \beta \times J} \rightarrow \mathbb{R}$ , and  $f_3 = \mathbb{R}^{\beta \times \beta \times I} \rightarrow \mathbb{R}$  functions are applied to each tract  $T_{xy}$ ,  $T_{xz}$  and  $T_{yz}$ .

$$I_{xy}^\beta(i, j) = f_1 \left( T_{xy}^\beta(i, j) \right) \quad (4)$$

$$I_{xz}^\beta(i, j) = f_2 \left( T_{xz}^\beta(i, j) \right) \quad (5)$$

$$I_{yz}^\beta(i, j) = f_3 \left( T_{yz}^\beta(i, j) \right) \quad (6)$$

$I_{xy}(i, j)$ ,  $I_{xz}(i, j)$ , and  $I_{yz}(i, j)$  are pixels with coordinates  $i, j$  of the  $XY$  projection,  $XZ$  projection and  $YZ$  projection respectively. Figure 2 shows the construction of these three projections.

Summarizing a video sequence in a single still frame that retains discriminant information is not an easy task. Ideally, this frame should retain the appearance information contained in the video sequence and it should contain dynamic information of the human action as well.

A scalar function will project a single view, whilst a vector function will project multiple views.

Even though, we have chosen the first four moments and the max function to project the views, any scalar or vector function can be used to do this. The moments are scalar quantities to characterize a function and to capture its significant features. On the other hand, supremum is a scalar function that it has been widely used for pooling in human action recognition. When we apply these functions to a dynamic video sequence, each one defines a different statistical feature of the motion. The projection triplet  $(I_{xy}, I_{xz}, I_{yz})$  condenses sequence information in three components and retains useful information for action recognition. We have computed the triplet from both components of the optical flow of video sequences instead of the raw sequence frames. We have considered  $Fx$  and  $Fy$  components of the optical flow of the video sequence and obtained two projection-triplets from the 3-modes fibers

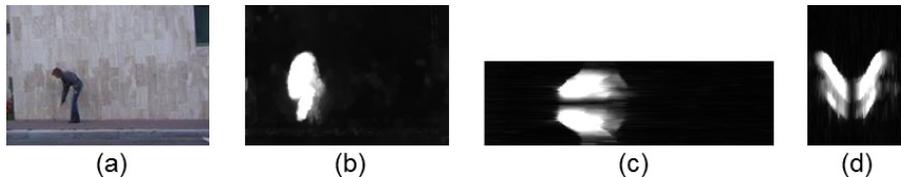


Figure 3: (a) Sample of the 'bend' action computed using the supremum as  $f_1$ ,  $f_2$ , and  $f_3$ , (b)  $I_{xy}$  projection, (c)  $I_{xz}$  projection (d)  $I_{yz}$  projection

of both tensors.  $I_{xy}$  image provides information about the relationship of the magnitude of the moving parts,  $I_{xz}$  provides information about relationship of magnitudes of horizontal motion and  $I_{yz}$  image provides information about relationship of magnitudes of vertical motion. Therefore,  $I_{xz}$  and  $I_{yz}$  views retain spatio-temporal information about motion of image objects, while  $I_{xy}$  retains information about motion appearance. Figure 3 shows a sample of the bend action from Weizmann dataset, and one of its projection triplets.

We have also computed a projection triplet for each color channel of the video sequence. We have chosen RGB color space for comparison purposes, since this is the one frequently used by other authors [41, 42, 11]. In color projections,  $I_{xy}$  captures the motion of the objects in the scene.  $I_{xz}$  and  $I_{yz}$  tend to capture changes in the pixel colour values in the  $XZ$  and  $YZ$  planes respectively. Depending on the  $f$  function used to compute the projections, different results are obtained. Figure 4 shows some examples of different projection functions applied to a RGB video sequence.

The choice of the projection function  $f$  is discussed in section 4.4. To take advantage from the complementarity of different  $f$  functions, we have also merged the results obtained from different templates projected using different  $f$  functions applying Sum Pooling, **Unlike [4], we have merged features obtained from 3 single projections computed using different functions (*max*, *mean*, *stdev*, *skewness* and *kurtosis*), while [4] computed different descriptors like HOG, HOF, MBH on each video frame, and merged the features obtained from these descriptors. HOG is a successful technique for recognizing static objects. Therefore, HOG adds complementary information to our features. HOF considers the amount of different directions of OF vectors, but the spatial relationship among these vectors is lost, instead the features computed on our triplet retain local information about the relationship of the moving parts. MBH is based on derivatives of OF in order to achieve robustness to camera motion. Our views use OF but not derivatives of OF, thereupon they**

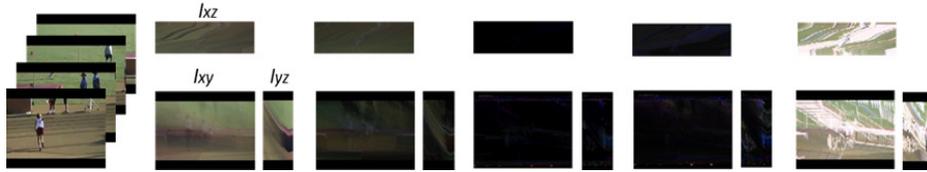


Figure 4: From left to right, (a) sample of the *HighJump* action of UCF101 dataset, (b) Projection using  $f = Mean$ , (c) Projection using  $sf = StDev$ , (d) Projection using  $f = Skewness$ , (e) Projection using  $f = Kurtosis$  and (f) Projection using  $f = Max$

provide different motion information. Finally, the  $I_{xy}$  and  $I_{yz}$  views are not considered in IDTs, and therefore it is a piece of complementary information added by our 3 views as well.

### 3.2. Single and Multiple tensor projection

Since a projection triplet  $(I_{xy}, I_{xz}, I_{yz})$  condenses sequence information in a single image,  $I_{xy}$  projection can lose important information in long sequences, especially in video sequences that contain more than one action. To avoid this inconvenience, we have experimented with two different scenarios.

#### Single view projection (*Sfunction*)

In a first scenario we compute a single image from each projection. Therefore, we have a triplet  $I_{xy}, I_{xz}, I_{yz}$  from each video sequence. Figures 3 and 4 show this scenario for OF and RGB video sequences respectively.

#### Multiple views projections (*Mfunction*)

In a second scenario, like suggested in [11], we generate multiple templates from each video sequence splitting them into multiple overlapping segments. We use temporal windows of size  $\tau$  overlapped  $\epsilon$  frames. The drawback is that we also increase the dataset size. Henceforward, the prefix  $M$  indicates that the projections were computed using multiple overlapping segments.

### 3.3. Local feature Extraction

Since  $I_{xy}, I_{xz}$  and  $I_{yz}$  are 2D images, the standard algorithms for keypoint detection and description for grey-level images can be used. We have considered four feature extractor/descriptors for our templates: PHOW, HOG, LIOP and SMFs.

Pyramid Histogram Of visual Words (PHOW) is a variant of the popular SIFT algorithm [19]. In SIFT, keypoints are first extracted from a set of reference images and their respective descriptors are stored in a database.

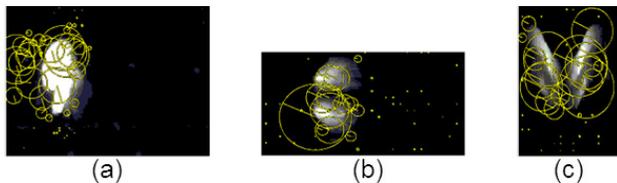


Figure 5: Sample of PHOW computed on bend human action for: (a) $I_{xy}$  projection, (b) $I_{xz}$  projection (c) $I_{yz}$  projection

Some feature descriptors like Dense SIFT or HOG do not present a detection stage, keypoints are computed over dense grids in the image domain rather than sparse interest points. This dense grid usually provides more information than the corresponding descriptors evaluated at a much sparser set of image keypoints. PHOW is a variant of Dense SIFT, extracted at multiple scales building a pyramid of descriptors. Once the pyramid is built, the descriptors are clustered using a  $k$ -means algorithm and each cluster is called a visual word. Authors are not aware that PHOW has been previously used for action recognition.

We extract keypoints from the projection triplets by means of dense sampling of the image using PHOW. Spatial coordinates of the obtained keypoints are accompanied by their scales, which define the extent of the neighborhood. The contents of the projected images ( $I_{xy}$ ,  $I_{xz}$  and  $I_{yz}$ ) are represented as a set of descriptors corresponding to the keypoints obtained. These descriptors, extracted from training sequences, are used to identify similar keypoints in new templates projected from new sequences. Figure 5 shows an example of the keypoints obtained using PHOW algorithm. Since PHOW computes a lot of keypoints, for visual purposes, we have only shown fifty keypoints selected randomly. Circles are centered in the keypoint locations, their radius are proportional to the scale, and the lines inside represent the main orientations. We can see that many keypoints are located on regions where the actions were performed and their orientations are very similar among them. Once all keypoints have been located at different scales, a SIFT descriptor is computed for each one (PHOW is simply a dense SIFT at several resolutions).

Other feature descriptors like HOGs, can be computed from the whole image without any previous keypoint detection. Figure 6 shows the HOG features obtained from  $I_{xy}$ ,  $I_{xz}$  and  $I_{yz}$ . We can see that a lot of gradient information is located on regions where the actions were performed.

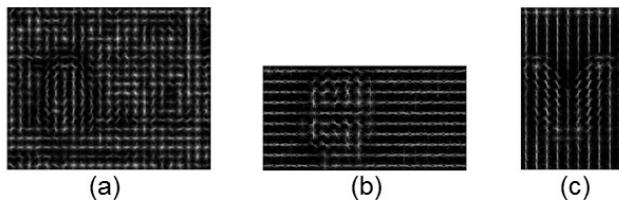


Figure 6: Sample of HOG computed on bend human action for: (a) $I_{xy}$  projection, (b) $I_{xz}$  projection (c) $I_{yz}$  projection

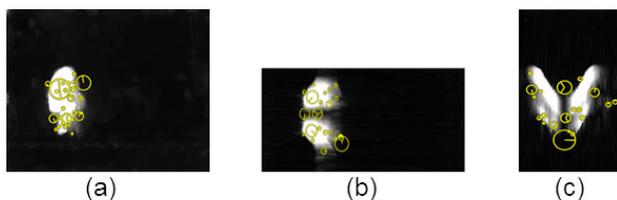


Figure 7: Sample of LIOP computed on bend human action for:(a) $I_{xy}$  projection, (b) $I_{xz}$  projection (c) $I_{yz}$  projection

Another feature descriptor, LIOP (Local Intensity Order Pattern ) [22], can also be obtained from our projections. LIOP can use different feature detection algorithms, thus, it can be combined with different feature detectors like Difference of Gaussians (DOG), Harris-Laplace, etc. The basic principle of LIOP descriptor is that the relative order of pixel intensities remains unchanged when the intensity changes are monotonic. First, the overall intensity order is used to divide the local patch into subregions called ordinal bins. Next, a LIOP is computed for each keypoint based on the relationships among the intensities of its neighbourhood. The LIOP descriptor is constructed by accumulating the LIOPs of the keypoints in each ordinal bin respectively, and stacking them together. Authors are not aware that LIOP has been used in action recognition.

Figure 7 shows the DoG features obtained in  $I_{xy}$  ,  $I_{xz}$  and  $I_{yz}$ . This figure shows that keypoints are located on regions where the actions were performed next, LIOP descriptors are obtained for each keypoint.

SMFs is a bioinspired feature extractor and descriptor. Some biological inspired techniques have obtained successful results in object recognition [40]. In these works, the model itself attempts to summarize a core of well-accepted facts about the ventral stream in the visual cortex: Visual processing is hi-

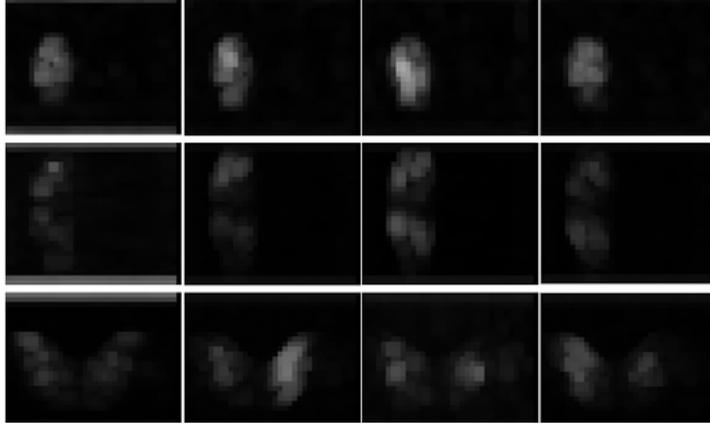


Figure 8: Sample of C1 SMFs features on bend action. Rows from top to bottom are  $I_{xy}$ ,  $I_{xz}$  and  $I_{yz}$  projections respectively. Columns from left to right correspond to -45, 0, 90 and 45 orientations of Gabor filters

erarchical, aiming to build invariance to position and scale first, and then, invariance to viewpoint and other transformations. Along the hierarchy, the receptive fields of the neurons, as well as the complexity of their optimal stimuli, increase. In its simplest form, SMFs consists of four layers of computational units, where simple  $S$  units alternate with complex  $C$  units. The  $S$  units combine their inputs with a bell-shaped tuning function to increase selectivity. The  $C$  units pool their inputs through a maximum operation, thereby increasing invariance. The approach consists of four layers of computational units  $S1$ ,  $C1$ ,  $S2$  and  $C2$  for object recognition. Bioinspired methods like SMFs can be also applied on our projections. Figure 8 depicts the  $C1$  features extracted in the training phase for SMFs. Each projection is represented by four images corresponding to four different orientations of Gabor filters. We can see the different response of each filter for a same region in the images.

#### 3.4. Feature representation

To represent the feature descriptors obtained in last section, we have experimented with two different feature representations, namely, Bag Of Features (BoF) and Fisher Vectors encoding ( $Fv$ ).

BoF approaches are characterized by the use of an orderless collection of image features, lacking any structure or spatial information. With this purpose, similar descriptors are clustered using a  $k$ -means algorithm. The

centers of these clusters define a Visual Codebook. The number of clusters is the size of the Visual Codebook and each cluster in the Visual Codebook forms a visual word. Once the dictionary is built, we compute a histogram for each training triplet. Each histogram models a human action. We have also built a randomized K-D tree forest of Visual Words to speed up the search.

In [37], they proved that  $Fv$  overcomes BoF for action recognition. The  $Fv$  extends BoF but it encodes the first and second order statistics of the features descriptors and a Gaussian Mixture Model (GMM).  $Fv$  describes the deviation of a set of descriptors from an average distribution of descriptors computed from a parametric generative model. We have studied also  $Fv$  to assess the performance improvement achieved in our templates. We have reduced the dimension of the descriptors using Principal Component Analysis (PCA) and applied  $l2$  normalization to the  $Fv$ .

### 3.5. Feature fusion and action classification

Since we obtain three templates from each component of the video sequence, we need to fuse the features extracted from each template to obtain the final classification. For BoF we have computed the features of each template and generated a single Visual Codebook. Then we have computed a histogram for each training triplet.

For  $Fv$  representation, we have computed a Gaussian Mixture Model (GMM) and obtained the  $Fv$  from each template. To fuse the features of the three templates, we have summed the power  $l2$ -normalized features of each projection.

Finally, for classifying the video sequence, we feed a Stochastic Dual Coordinate Ascent Methods (SDCA) linear SVM solver with BoF or  $Fv$  features. It is an effective technique for solving regularized loss minimization problems in machine learning.

In a first stage, we have considered only the features obtained using a single function to compute the templates, but there is no reason for not using multiple functions and fuse the features obtained to improve the performance. We have computed different templates using *max*, *mean*, *Stdev*, *skewness* and *kurtosis* functions with  $Fv$  representation and fused the features obtained from each one summing the power  $l2$ -normalized features of each template.

Our templates can be considered as a motion feature descriptor like HOF and MBH. Unlike, HOF and MBH that compute dense features in each frame considering the  $XY$  plane, we compute features in  $XY$ ,  $XZ$  and  $YZ$  planes.

We have assessed the performance improvement brought when we mix the obtained features with IDTs. To fuse our features with IDTs, we tested representation and score level and we obtained the best results using score level [27]. We think that score level performs better than representation level because the features obtained from the projected templates are independent of IDTs features as suggested in [27].

## 4. Experimental Setup

### 4.1. Objective of the experiments

In this section we show that the presented method based on the tensor projected has a good performance in action recognition by itself, and it is also complementary to IDTs, improving its performance when the descriptors of both techniques are fused together. To prove it, we should compare the results obtained, with the ones published by other authors in recent literature using the same datasets. In order to test the performance of the system, once it has been trained, we will measure the recognition rate for the test sequences using the simple ratio:

$$RR = \frac{\# \text{ samples correctly classified}}{\text{Total samples Tested}} \quad (7)$$

First, we will determine the optimum tuning parameters needed for obtaining the templates and descriptors. Once the whole process is correctly tuned, we will combine it with IDTs. The first problem that arises is that different projection functions or keypoint descriptors will lead to totally different templates. Thus, the first objective of the experiments should be to test different functions  $f$  used to project the subtensors, and to determine the tract width  $\beta$ , the keypoint detector/descriptor, and the feature representation to be used.

With the objective of tuning all the process parameters for obtaining the highest recognition rate, we have designed a first experiment. In this experiment, we have computed multiple  $(I_{xy}, I_{xz}, I_{yz})$  triplets projecting them using five different functions (*supremum*, *mean*, *standard deviation*, *skewness* and *kurtosis*). We have set  $f1 = f2 = f3$  for all tests. Moreover we have used six different tract values ( $\beta=0, 2, 4, 6, 8$  and  $10$ ). We have also tested the performance of the system using four different keypoint detectors/descriptors (PHOW, LIOP, HOG and SMFs) for the 30 triplets obtained.  $RR$  is computed for every projection function, every neighbourhood

$\beta$ , and every keypoint descriptor. Once we have determined the optimal  $\beta$  and extractor/descriptor, we have fixed them for the rest of the experiments.

In a second experiment, we have evaluated if  $Fv$  feature representation improves the recognition rate with respect to BoW.

We have also compared the performance using single or multiple tensor projection in a third experiment.  $Sfunction$  computes a single image from each projection whilst  $Mfunction$  generates multiple templates from each video sequence splitting them into multiple overlapping segments.

In a fourth experiment we have studied the complementarity of the five projection functions *supremum*, *mean*, *standard deviation*, *skewness* and *kurtosis*, merging them using sum pooling, and evaluating the improvement of the performance achieved when each one is added to the classifier.

Once all the parameters have been fixed, we have combined our method with IDTs and assessed the improvement of performance provided by our templates. Finally, we compare the results against other state-of-the-art techniques on the four public datasets used. With this purpose, we have fixed all the parameters to the ones that gave the best RR in the previous experiments.

#### 4.2. Datasets

With the aim of tuning the parameters of our algorithms and comparing the results with other author’s proposals, we have used four publicly available human actions datasets, namely, Weizmann [47], KTH [48], UFC101 [49] and HMDB51 [50]. These datasets have been widely used in human action and gesture recognition. Weizmann and KTH datasets contain simple action sequences; therefore, they are very useful for tuning the system. On the other hand, UCF101 and HMDB51 present more complex human action sequences; they are more challenging and will be used for comparison purposes.

**Weizmann** dataset has been recorded with static camera and static background; there aren’t occlusions and there is just a single object moving in each sequence. It doesn’t present important illumination changes. This dataset is composed by 10 different actions made by 9 different persons.

In **KTH** dataset there is just a single object moving in each sequence, there aren’t occlusions but there are scale variations, and people wearing different clothes. It presents 600 indoor and outdoor video sequences. This dataset is composed by six types of human actions made by 25 persons.

**UCF101** dataset is composed by 101 human action classes. It has 13,320 video clips with 320 x 240 pixels resolution frames. Each video contains a

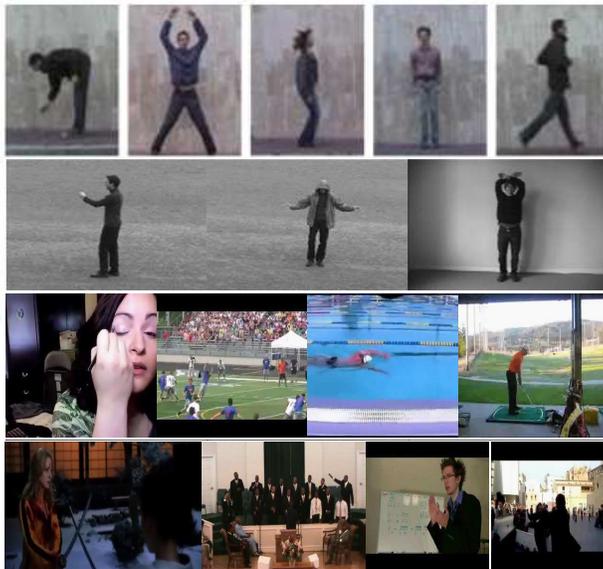


Figure 9: Some examples of human actions used in the experiments. Rows from top to bottom are examples of Weizmann, KTH, UCF101 and HMDB51 datasets respectively.

single action in different environments and moderate background clutter.

**HMDB51** is the dataset containing the most complex sequences. It contains 51 human actions in 6766 video clips. The sequences contain a single action in realistic scenarios downloaded from YouTube. There are scale and illumination changes.

Some examples of some actions taken from these four datasets are shown in Figure 9.

### 4.3. Experimental results

In this section, we present and discuss the experimental results of the evaluation using the datasets shown in last section. For further reproducibility of the results, we detail next all tuning parameters of the algorithms used.

For PHOW we have used  $\tau = 1$ ,  $\rho = 3$ , where  $\tau$  is the distance between keypoints and  $\rho$  is the descriptor size. We have tested codewords from 100 to 1100 visual words for PHOW and LIOP. For HOG we have tested from 9 to 36 bins and variable cells from 3 to 15 pixels. We have used 1000 patches and 4 orientation filters (-45, 0, 90, 45) for SMFs.

For  $Fv$ , we have tested our system using *max* and *standard deviation* functions in RGB and OF video sequences individually, and PHOW as descriptor.

For GMM training, we have sampled 256,000 features chosen randomly, ranging from 16 to 512 the GMM size. We have also tested  $Fv$  and PCA-Whiten with respect traditional PCA. We tested also power  $l2$ -normalization with respect power  $l1$ -normalization.

The stage where the 3 views are going to be fused is an important decision, because it will affect to the final performance. Intuitively we can see that the three 3 views are not related in the cuboid level, since our method considers the whole XYZ coordinates from the video sequence, not XYZ local regions like cuboids. Fusing the three views in representation level could be a good choice because the features are correlated in video level. Score level is not a good choice because the features are not independent. To confirm these hypotheses, we have experimented the three fusion methods and the results have confirmed that representation-level is the best fusion method for our templates.

Finally we have tested Max Pooling with respect Sum Pooling to fuse the features generated from different functions and to fuse them with IDTs.

We obtained the best results using 256 Codebook size, PCA-Whiten performed better than traditional PCA, power  $l2$ -normalization obtained better results than power  $l1$ -normalization and Sum Pooling improved the results both for templates feature fusion and for IDTs feature fusion. Therefore, this has been the configuration used for the rest of experiments.

All results and data from the state-of-the-art techniques used for comparison purposes, have been taken from the authors original papers.

For the Weizmann and KTH datasets, we have used leave-one-out cross validation method for testing because this is the method normally used by other authors for these datasets. For the UCF101 and HMDB51 datasets, we have performed the evaluations according the three train/test splits released by their authors.

The values of the parameters used for tuning our technique are fixed according to the results of the experiments carried out.

#### *4.4. Keypoint descriptor and parameter selection*

For the Weizmann dataset, 10 actions made by one person are used for test and the actions made by the remaining 8 persons are used for training. We repeat this process for each one of the 9 persons. At the end, we have the recognition rate for each one of the 9 persons for each one of the 10 actions. For the KTH dataset, 6 actions made by one person are used for test and the actions made by the remaining 24 persons are used for training. We

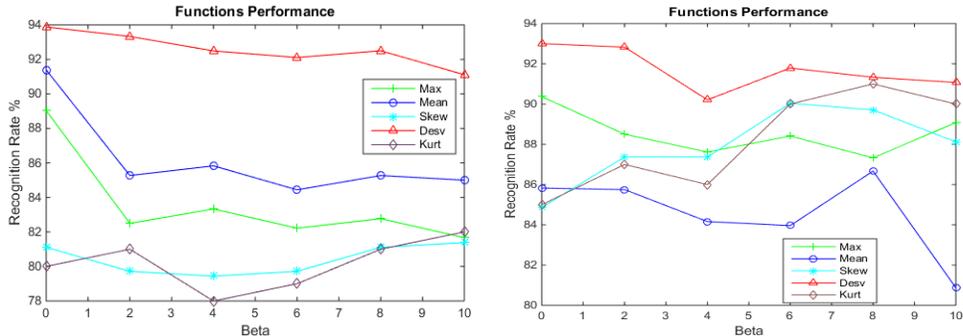


Figure 10: Average performance of PHOW, LIOP, HOG and SMFs descriptors for different projection functions respect to  $\beta$  for Weizmann (left) and KTH (right) dataset.

repeat this process for each one of the 25 persons, and at the end we get the recognition rate for each person and action. Then, we compute the mean recognition rate,  $\overline{RR}$ .

This process is repeated using five different projection functions (*Supremum*, *Mean*, *StDev*, *Skewness* and *Kurtosis*), six different tract neighbourhoods ( $\beta=0, 2, 4, 6, 8$  and  $10$ ), and four different keypoint descriptors (PHOW, LIOP, HOG and SMFs). Figures 10 and 11 show the average results. Figure 10, shows the average performance for each function with respect to  $\beta$  value. We can see that using the standard deviation of fiber values, the minimum average recognition rate is above 91% and the best one is close to 94% and 93% for Weizmann and KTH datasets respectively. Results show that the results obtained with the *StDev* projection function outperforms the rest of functions in most cases. Results show also that a tract  $\beta=0$  gives the best average performance.

Figure 11 shows the average performance of each feature descriptor with respect to  $\beta$ . Results show that PHOW performs better than the rest of techniques, for all  $\beta$  values.

Our best results were, 98.8% classification rate on Weizmann dataset using *StDev* as projection function, PHOW descriptor,  $\beta=0$  and 900 visual words. For KTH we obtained 97.5% classification rate using *StDev* function, PHOW descriptor,  $\beta$  between 2 and 4, and 900 visual words.

Since, global results show that the improvement achieved using  $\beta > 0$  is not significant, we will use PHOW as keypoint descriptor and  $\beta=0$  for the rest of the experiments.

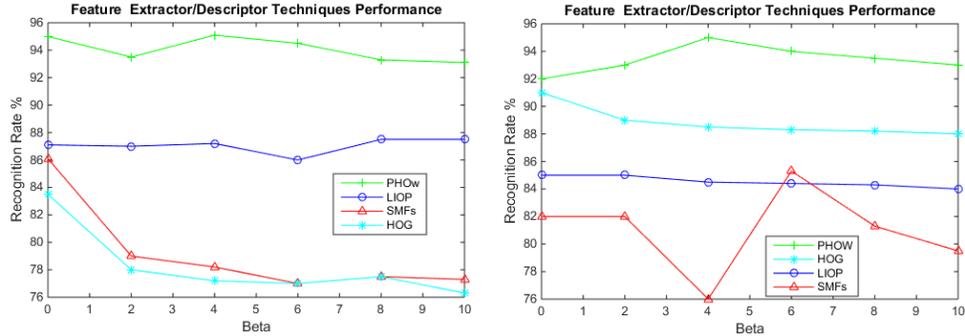


Figure 11: Average performance of  $Max$ ,  $Mean$ ,  $StDev$ ,  $Skew$  and  $Kurt$  functions for each extractor/descriptor technique respect to  $\beta$  for Weizmann (left) and KTH (right) datasets.

#### 4.5. Feature representation evaluation

We have evaluated whether the  $Fv$  representation improves the results with respect BoW representation on UCF101 dataset. For this experiment, we have not used the simplest datasets because the high recognition rates obtained make difficult to perceive the difference of performances between BoW and  $Fv$ .

Tables 1 and 2 show the results obtained for each  $I_{xy}$ ,  $I_{xz}$  and  $I_{yz}$  projections individually and combined, projecting RGB or optical flow channels. We have studied the performance obtained for each projection individually and the 3 projections combined summing the power  $l_2$ -normalized features of each projection. Table 1 shows the results using  $Fv$  feature representation and Table 2 the results using BoW feature representation. We conclude that Fisher vector representation is more suited for working with our projections, therefore we will use Fisher vector representation for the rest of experiments.

#### 4.6. Single view projection vs Multiple view projections

In this experiment we are interested in determining if it is justified to compute  $Mfunction$  instead of  $Sfunction$ .

We have computed  $Sfunction$  and  $Mfunction$  on RGB and OF video sequences using the UCF 101 dataset. Table 3 shows the results obtained for each split and the average result for  $Sfunction$  using UCF101. To compute  $Mfunction$ , we have generated multiple templates from each video sequence splitting them into multiple overlapping segments. We use temporal windows of size 15 overlapped 5 frames.

Table 1: Results for  $I_{xy}$ ,  $I_{xz}$  and  $I_{yz}$  individually and combined on UCF101 dataset for RGB a OF video sequences using  $Fv$ .

$Fv$				
Function	$I_{xy}$	$I_{xz}$	$I_{yz}$	Combined
$S_{mean}$ (RGB)	49.52	42.58	48.24	63.25
$S_{StDev}$ (RGB)	50.31	44.32	50.67	65.61
$S_{skew}$ (RGB)	26.47	24.85	24.9	38.31
$S_{kurt}$ (RGB)	24.58	29.1	27.58	40.12
$S_{max}$ (RGB)	47.2	40.15	47.5	62.41
$S_{mean}$ (OF)	34.11	53.07	52.85	61.02
$S_{StDev}$ (OF)	50.41	52,04	51,88	64,88
$S_{skew}$ (OF)	25.36	23.21	22.10	40.52
$S_{kurt}$ (OF)	23.57	26.98	27.69	41.25
$S_{max}$ (OF)	39.9	55.37	59.97	67.93

Table 2: Results for  $I_{xy}$ ,  $I_{xz}$  and  $I_{yz}$  individually and combined on UCF101 dataset for RGB a OF video sequences using  $BoW$ .

$BoW$				
Function	$I_{xy}$	$I_{xz}$	$I_{yz}$	Combined
$S_{mean}$ (RGB)	25.24	26.3	27.7	35.21
$S_{StDev}$ (RGB)	26.1	27.2	28.9	37.8
$S_{skew}$ (RGB)	18.14	17.14	16.4	25.14
$S_{kurt}$ (RGB)	17.25	18.54	18.2	23.8
$S_{max}$ (RGB)	26	28.67	27.6	34.57
$S_{mean}$ (OF)	26.7	27.4	24.1	34.01
$S_{StDev}$ (OF)	27,51	28,3	30,1	40,3
$S_{skew}$ (OF)	18.9	17.54	19.2	27.02
$S_{kurt}$ (OF)	19.1	18.4	16.4	27.23
$S_{max}$ (OF)	28.7	28.9	32.4	42.1

Table 4 shows the results obtained for each  $f$  function used. We have discarded the features generated using *Kurtosis* and *Skewness* functions because they got a bad performance in the previous experiment. We can see that *Mfunction* overcomes *Sfunction* in all functions both in RGB and OF video sequences.

Table 3: Results of  $Sfunction$  methods for RGB and OF video sequences on the UCF101 dataset.

Method	SPLIT1	SPLIT2	SPLIT3	Average
$S_{mean}(RGB)$	63.25	62.45	63.85	63.18
$S_{StDev}(RGB)$	65.61	66.25	64.85	65.57
$S_{max}(RGB)$	62.41	61.89	63.58	62.62
$S_{mean}(OF)$	61.02	60.24	62.35	61.2
$S_{StDev}(OF)$	64.88	65.2	64.8	64.96
$S_{max}(OF)$	67.93	65.4	66.87	66.73

Table 4: Results of  $Mfunction$  methods for RGB and OF video sequences on the UCF101 dataset.

Method	SPLIT1	SPLIT2	SPLIT3	Average
$M_{mean}(RGB)$	69.15	70.54	69.84	69.85
$M_{StDev}(RGB)$	70.6	73.35	72.35	72.1
$M_{max}(RGB)$	69.38	66.99	68.53	68.3
$M_{mean}(OF)$	75.01	72.5	75.69	74.4
$M_{StDev}(OF)$	69.88	71.28	71.66	70.94
$M_{max}(OF)$	74.89	75.14	76.56	75.53

#### 4.7. Contribution of different $f$ functions

In this experiment we want to determine how the performance is improved when we combine the templates projected using different  $f$  functions. We have combined each  $f$  function using Sum Pooling. To do this, we have summed the power  $l_2$ -normalized features of each function. Table 5 shows the improvement of performance obtained when we add each one of the templates computed from a different  $f$  function. We have used Sequential Forward Selection (SFS)[51] to add the functions progressively. In this way, we have selected first the function that obtained the best result individually ( $M_{max}(OF)$ ). For the rest of the features, we have added the feature that results in the highest performance when combined with the previous features that have already been selected. Moreover, alike previous experiment, we have discarded the features generated using *Kurtosis* and *Skewness* functions because they got worse performance when they were combined with other previously selected features.

Table 5: Significance of templates on UCF101 and HMDB51 datasets.

Template	UCF101	HMDB51
$M_{max}(OF)$	75.53	41.1
$+M_{mean}(OF)$	76.83	43.26
$+M_{StDev}(OF)$	78.14	44.11
$+M_{StDev}(RGB)$	79.08	45.2
$+M_{mean}(RGB)$	79.81	45.9
$+M_{max}(RGB)$	80.1	46.2

Table 6: Recognition Rate (%) obtained by IDTs feature descriptors independently, templates and combining IDTs and our templates on UCF101 and HMDB51 datasets.

Method	UCF101	HMDB51
<i>HOG</i>	74.06	45.9
<i>HOF</i>	75.3	45.3
<i>MBHx</i>	77.41	45.25
<i>MBHy</i>	78.12	47.2
$M_{max+mean+StDev}(OF\&RGB)$	80.1	46.2
$HOG + HOF + MBHx + MBHy(IDTs)$	87	60
$IDTs + M_{max+mean+StDev}(OF\&RGB)$	89.3	65.3

We can see from top to bottom the improvement of performance added by each template. Although OF sequences present a higher performance than RGB sequences, we can see that the three RGB functions summed achieve an improvement of 2% approximately.

We conclude that we can improve the performance combining different  $f$  functions computed on RGB and OF video sequences.

#### 4.8. Fusion with IDTs

So far, we have studied how to obtain the best performance of our method based on templates. Since one of the goals of this paper is improve IDTs performance, we have added our descriptors to IDTs using the configuration that obtained the best results in the previous experiments, and we have assessed the improvement on the performance. First, we have compared the performance obtained by each feature descriptor (HOG, HOF, MBH and the ones extracted from the templates) individually. Then, we have fused all

Table 7: Recognition Rate (%) on the Weizmann and KTH datasets.

Weizmann	%	KTH	%
Scovanner et al. [25]	84.2	PM [17]	97
Jhuang et al. [43]	98,8	TCCA [14]	95
Bregonzio et al. [52]	96.66	PT [53]	93,4
		Bregonzio et al. [52]	94.33
		Liu et al. [54]	95.5
<b>Ours</b>	<b>98.8</b>		<b>97.5</b>

feature descriptors and compared its performance against the original IDTs. We have fused all the different descriptors using score level fusion [27]. For both techniques we have used Fisher encoding [27]. We have extracted HOG, HOF, MBH, and trajectory features from the video sequences using the code supplied by the original authors and created GMMs of size 256. Next, we have reduced the dimensionality of each descriptor applying PCA with a factor of 0.5. The dimensions of the descriptors have been: 30 for Trajectories, 96 for HOG, 108 for HOF and 192 for MBH.

Table 6 shows the performance obtained by each feature descriptor individually and the result obtained when we combine them. We can see that the results using templates overcome HOG, HOF and MBH components in most of cases. Moreover, we can see that our features are complementary of IDTs. We obtained 87% and 60% using only IDTs in UCF101 and HMDB51 datasets respectively. Therefore, our templates bring an improvement of 2% and 5% for UCF101 and HMDB51 datasets respectively

#### 4.9. Comparison with other action recognition techniques

We have compared our method against other state-of-the-art techniques. We showed in the chapter 4.4 that the results obtained on simple datasets as Weizmann and KTH were very high. Therefore, we consider that computing IDTs, multiple functions and  $Fv$  is not justified for these datasets, since the results obtained using only descriptors with a single function like  $Stdv$  and BoW feature representation are enough to outperform the state-of-the-art. The experiments with these simple datasets have been included just for comparison purposes with those methods that publish they results using only on these datasets.

For more complex datasets like UCF101 and HMDB51, we have used the

Table 8: Recognition Rate (%) on the UCF101 and HMDB51 datasets.

UCF101	%	HMDB51	%
Shallow			
Wu et al. [37]	84,2	Fernando et al.[8]	63,7
Peng et a.l [27]	87,9	Hoai et al.[12]	60,8
Lan et al.[55]	89,1	Peng et al. [56]	66,8
Peng et al. [36]	87,5	Peng et al. [27]	61,9
		Lan et al.[55]	65,4
Ours	<b>89.3</b>		65.3
deep			
Bilen et al.[11]	89,1	Bilen et al.[11]	65,2
Yue-Hei-Ng et al. [41]	88,6	Simonyan et al.[42]	59,4
Simonyan et al.[42]	88		

whole technique (IDTs + templates) to compare our technique against other state-of-the-art ones. Tables 7 and 8 show the comparative results of our method. Table 7 shows the results for two simple datasets as Weizmann and KTH and Table 8 the results for more complex datasets as UCF101 and HMDB51.

On the other hand, for UCF101 and HMDB51 datasets we use the IDTs+ $M_{max+mean+StDev}(OF\&RGB)$  combination and  $Fv$  feature representation. Table 8 shows deep techniques (CNN) and shallow techniques (no CNN) separately.

The results show that a single function and BoW is enough to overcome most state-of-the-art techniques in simple datasets. The results show also that our method overcomes most state-of-the-art techniques in more complex datasets when we combine OF and RGB templates with  $Fv$  representation.

For UCF101 and HMDB51 datasets, only [41] and [42] don't use IDTs in their finals results, the rest of authors use or combine their techniques with IDTs. Although CNN techniques like [41] and [42] obtain very good results without using IDTs, we think that our method is only directly comparable with Bilen [11], because the rest of CNN methods compute features in each frame of the video sequences. Bilen studied two different scenarios, namely Single Dynamic Image (SDI) and Multiple Dynamic Image (MDI). These ones are similar to our  $Sfunction$  and  $Mfunction$  scenarios, and therefore,

they can be compared directly. For SDI they tested *Max* and *Mean* functions, and their proposed Dynamic Images. They obtained 45.4%, 52.6% and 57.9% respectively, whilst we have obtained 62.62% and 63.18% on the *Sfunction* scenario for *Max* and *Mean* functions respectively. Our best results have been 65.57% using *Standard Deviation* function on RGB sequences and 66.73% using *Max* function on OF sequences. **Bilen considers only the  $I_{xy}$  view of the video sequences to represent human actions. If we compare our results using only  $I_{xy}$  view against Bilen ones, both results are very similar, but when we add  $I_{xz}$  and  $I_{yz}$ , our method overcomes Bilen’s widely.** For MDI, Bilen obtained 70.9% using Dynamic Image for the UCF101 dataset, whilst we have obtained 72.1% on the *Mfunction* scenario using *Standard Deviation* function on RGB video sequences and 75.53% using *Max* function on OF video sequences. Even though learned features (CNN) could obtain better results than ours when large datasets are available, engineered features can be still useful when few training samples are available as proved in [57]. Moreover, engineered features like ours are still useful because they can improve the performance when combined with CNNs features, as suggested in [58].

## 5. Conclusions

We have presented a view-based temporal template for action recognition. We compute temporal templates projecting RGB video sequences and both components of optical flow using simple functions as *Standard Deviation*, *Supremum*, *Mean*, *Skewness* and *Kurtosis*. Standard feature descriptor techniques are used on the projections obtained. PHOW performed better than the rest feature descriptors on our templates.

We have experimented also with simple and multiple view scenarios. *Mfunction* computes multiple templates from each video sequence splitting them into multiple overlapping segments whilst *Sfunction* computes a single image from each projection. *Mfunction* obtained better performance than *Sfunction*.

Experimental results show also that *Fv* encoding gives more accurate results than a Bag of Features (BoF) approach.

The features obtained using different projection functions like *Supremum*, *Mean* and *Standard Deviation* combined using sum pooling are complementary and the performance increases when more functions are added.

The descriptors obtained from our templates obtain a very good performance when they are compared against HOG, HOF or MBH individually. The method presented based on tensor projections, reduces considerably the amount of data to be processed. It is simple and effective, and overcomes the results obtained by other action recognition methods on simple datasets. In order to achieve state-of-the-art recognition rates, we merge the features so obtained with the ones of IDTs. Using the templates features as a complement of HOGs, HOFs, and MBH, overcomes the rest of techniques in more complex datasets like UCF101 and HMDB51, improving the performance by 2% and 5% respectively.

## 6. Further work

In recent years, most of authors combine their CNNs techniques with IDTs features to improve the results. As suggested in [58], engineered features are still useful because they can improve the performance when combined with CNNs features. In the future we will test if our features are complementary with CNNs features to improve the state-of-the-art results.

We are also currently working on improving the performance of our method using differential OF, multiples scales in PHOW or other encoding methods like [55] and [56].

On the other hand, **even though we have chosen simple functions like the first four moments and supremum function to compute the templates, nothing prevents to use more complex functions.** Increasing the performance ratio by means of adding new complementary projection functions remains as an open research issue.

## Acknowledgements

This research was supported by the CICYT research project DPI2016-78957-R(AEI/FEDER, EU)

## References

- [1] R. Poppe, A survey on vision-based human action recognition, *Image and Vision Computing* 28 (6) (2010) 976 – 990.
- [2] S. Herath, M. Harandi, F. Porikli, Going deeper into action recognition: A survey, *Image and Vision Computing* 60 (2017) 4 – 21.

- [3] H. Wang, A. Klser, C. Schmid, C. L. Liu, Action recognition by dense trajectories, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2011, pp. 3169–3176.
- [4] H. Wang, C. Schmid, Action recognition with improved trajectories, in: IEEE International Conference on Computer Vision (ICCV), 2013, pp. 3551–3558.
- [5] P.-H. Gosselin, N. Murray, H. Jgou, F. Perronnin, Revisiting the fisher vector for fine-grained classification, *Pattern Recognition Letters* 49 (2014) 92 – 98.
- [6] V. Christlein, D. Bernecker, F. Hnig, A. Maier, E. Angelopoulou, Writer identification using gmm supervectors and exemplar-svms, *Pattern Recognition* 63 (2017) 258 – 267.
- [7] A. F. Bobick, J. W. Davis, The recognition of human movement using temporal templates, *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 23 (3) (2001) 257–267.
- [8] B. Fernando, E. Gavves, M. J. Oramas, A. Ghodrati, T. Tuytelaars, Modeling video evolution for action recognition, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 5378–5387.
- [9] H. Meng, N. Pears, M. Freeman, C. Bailey, Motion history histograms for human action recognition, in: *Embedded Computer Vision*, 2009, pp. 139–162.
- [10] E. P. Ijjina, K. M. Chalavadi, Human action recognition in rgb-d videos using motion sequence information and deep learning, *Pattern Recognition* 72 (2017) 504 – 516.
- [11] H. Bilen, B. Fernando, E. Gavves, A. Vedaldi, S. Gould, Dynamic image networks for action recognition, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 3034–3042.
- [12] M. Hoai, A. Zisserman, Improving human action recognition using score distribution and ranking, in: 12th Asian Conference on Computer Vision (ACCV), 2015, pp. 3–20.

- [13] M. S. Ryoo, B. Rothrock, L. Matthies, Pooled motion features for first-person videos, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 896–904.
- [14] T.-K. Kim, R. Cipolla, Canonical correlation analysis of video volume tensors for action categorization and detection, IEEE Transactions on Pattern Analysis and Machine Intelligence 31 (8) (2009) 1415–1428.
- [15] W. Liu, X. Yang, D. Tao, J. Cheng, Y. Tang, Multiview dimension reduction via hessian multiset canonical correlations, Information Fusion 41 (2018) 119 – 128.
- [16] D. Tao, X. Li, X. Wu, S. J. Maybank, General tensor discriminant analysis and gabor features for gait recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 29 (10) (2007) 1700–1715.
- [17] Y. M. Lui, J. Beveridge, M. Kirby, Action classification on product manifolds, in: Computer Vision and Pattern Recognition (CVPR), IEEE Conference on, 2010, pp. 833–839.
- [18] P. Loncomilla, J. R. del Solar, L. Martnez, Object recognition using local invariant features for robotic applications: A survey, Pattern Recognition 60 (2016) 499 – 514.
- [19] D. G. Lowe, Distinctive image features from scale-invariant keypoints, International Journal of Computer Vision. 60 (2) (2004) 91–110.
- [20] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: IEEE Computer Society Conference on Computer Vision and Pattern Recognition(CVPR’05), Vol. 1, 2005, pp. 886–893.
- [21] H. Meng, N. Pears, Descriptive temporal template features for visual motion recognition, Pattern Recognition Letters 30 (12) (2009) 1049 – 1058, image/video-based Pattern Analysis and HCI Applications.
- [22] Z. Wang, B. Fan, F. Wu, Local intensity order pattern for feature description, in: International Conference on Computer Vision (ICCV), 2011, pp. 603–610.
- [23] M. Heikkil, M. Pietikinen, C. Schmid, Description of interest regions with local binary patterns, Pattern Recognition 42 (3) (2009) 425 – 436.

- [24] M. Villamizar, J. Andrade-Cetto, A. Sanfeliu, F. Moreno-Noguer, Bootstrapping boosted random ferns for discriminative and efficient object classification, *Pattern Recognition* 45 (9) (2012) 3141 – 3153, best Papers of Iberian Conference on Pattern Recognition and Image Analysis (IbPRIA'2011).
- [25] P. Scovanner, S. Ali, M. Shah, A 3-dimensional sift descriptor and its application to action recognition, in: *Proceedings of the 15th international conference on Multimedia, 2007*, pp. 357–360.
- [26] T. R. Almaev, M. F. Valstar, Local gabor binary patterns from three orthogonal planes for automatic facial expression recognition, in: *Proceedings of the 2013 Humaine Association Conference on Affective Computing and Intelligent Interaction (ACII '13)*, 2013, pp. 356–361.
- [27] X. Peng, L. Wang, X. Wang, Y. Qiao, Bag of visual words and fusion methods for action recognition, *Comput. Vis. Image Underst.* 150 (C) (2016) 109–125.
- [28] Y. Kong, X. Zhang, W. Hu, Y. Jia, Adaptive learning codebook for action recognition, *Pattern Recognition Letters* 32 (8) (2011) 1178 – 1186.
- [29] N. Dalal, B. Triggs, C. Schmid, Human detection using oriented histograms of flow and appearance, in: *9th European Conference on Computer Vision (ECCV)*, 2006, pp. 428–441.
- [30] J. Zhang, M. Marszałek, S. Lazebnik, C. Schmid, Local features and kernels for classification of texture and object categories: A comprehensive study, *International Journal of Computer Vision* 73 (2) (2007) 213–238.
- [31] L. Liu, L. Wang, X. Liu, In defense of soft-assignment coding, in: *International Conference on Computer Vision (ICCV)*, 2011, pp. 2486–2493.
- [32] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, Y. Gong, Locality-constrained linear coding for image classification, in: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CPVR' 2010)*, 2010, pp. 3360–3367.

- [33] M. Sekma, M. Mejdoub, C. B. Amar, Human action recognition based on multi-layer fisher vector encoding method, *Pattern Recognition Letters* 65 (2015) 37 – 43.
- [34] H. Jgou, F. Perronnin, M. Douze, J. Snchez, P. Prez, C. Schmid, Aggregating local image descriptors into compact codes, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34 (9) (2012) 1704–1716.
- [35] X. Zhou, K. Yu, T. Zhang, T. S. Huang, Image classification using super-vector coding of local image descriptors, in: *11th European Conference on Computer Vision (ECCV)*, 2010, pp. 141–154.
- [36] X. Peng, L. Wang, Z. Cai, Y. Qiao, Q. Peng, Hybrid super vector with improved dense trajectories for action recognition, in: *International Conference on Computer Vision (ICCV Workshops)*, Vol. 13, 2013.
- [37] J. Wu, Y. Zhang, W. Lin, Towards good practices for action video encoding, in: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 2577–2584.
- [38] H.-J. Jung, K.-S. Hong, Modeling temporal structure of complex actions using bag-of-sequencelets, *Pattern Recognition Letters* 85 (2017) 21 – 28.
- [39] W. Liu, Z. J. Zha, Y. Wang, K. Lu, D. Tao,  $p$ -laplacian regularized sparse coding for human activity recognition, *IEEE Transactions on Industrial Electronics* 63 (8) (2016) 5120–5129.
- [40] T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber, T. Poggio, Robust object recognition with cortex-like mechanisms, *IEEE Transactions in Pattern Analysis and Machine Intelligence* 29 (3) (2007) 411–426.
- [41] J. Y.-H. Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, G. Toderici, Beyond short snippets: Deep networks for video classification, in: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 4694–4702.
- [42] K. Simonyan, A. Zisserman, Two-stream convolutional networks for action recognition in videos, in: *Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS’14)*, 2014, pp. 568–576.

- [43] H. Jhuang, T. Serre, L. Wolf, T. Poggio, A biologically inspired system for action recognition, in: IEEE 11th International Conference on Computer Vision (ICCV), 2007, pp. 1–8.
- [44] Z. Zhao, H. Ma, X. Chen, Semantic parts based top-down pyramid for action recognition, *Pattern Recognition Letters* 84 (2016) 134 – 141.
- [45] C. Xu, D. Tao, C. Xu, A survey on multi-view learning, CoRR abs/1304.5634. arXiv:1304.5634 (2013).
- [46] M. Ma, N. Marturi, Y. Li, A. Leonardis, R. Stolkin, Region-sequence based six-stream cnn features for general and fine-grained human action recognition in videos, *Pattern Recognition* 76 (2018) 506 – 521.
- [47] M. Blank, L. Gorelick, E. Shechtman, M. Irani, R. Basri, Actions as space-time shapes, in: Tenth IEEE International Conference on Computer Vision (ICCV’05), Vol. 2, 2005, pp. 1395–1402.
- [48] C. Schuldt, I. Laptev, B. Caputo, Recognizing human actions: a local svm approach, in: Proceedings of the 17th International Conference on Pattern Recognition (ICPR), Vol. 3, 2004, pp. 32–36.
- [49] K. Soomro, A. R. Zamir, M. Shah, Ucf101: A dataset of 101 human actions classes from videos in the wild, Computing Research Repository. (2012) abs/1212.0402.
- [50] H. Kuehne, H. Jhuang, R. Stiefelhagen, T. Serre, Hmdb51: A large video database for human motion recognition, in: Transactions of the High Performance Computing Center (HLRS), 2013, pp. 571–582.
- [51] A. W. Whitney, A direct method of nonparametric measurement selection, *IEEE Transactions on Computers* C-20 (9) (1971) 1100–1103.
- [52] M. Bregonzio, T. Xiang, S. Gong, Fusing appearance and distribution information of interest points for action recognition, *Pattern Recognition* 45 (3) (2012) 1220 – 1234.
- [53] Z. Lin, Z. Jiang, L. S. Davis, Recognizing actions by shape-motion prototype trees, in: IEEE 12th International Conference on Computer Vision (ICCV), 2009, pp. 444–451.

- [54] L. Liu, L. Shao, P. Rockett, Boosted key-frame selection and correlated pyramidal motion-feature representation for human action recognition, *Pattern Recognition* 46 (7) (2013) 1810 – 1818.
- [55] Z.-Z. Lan, M. Lin, X. Li, A. G. Hauptmann, B. Raj, Beyond gaussian pyramid: Multi-skip feature stacking for action recognition., in: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 204–212.
- [56] X. Peng, C. Zou, Y. Qiao, Q. Peng, Action recognition with stacked fisher vectors, in: *13th European Conference on Computer Vision (ECCV)*, 2014, pp. 581–595.
- [57] C. Shi, Y. Wang, F. Jia, K. He, C. Wang, B. Xiao, Fisher vector for scene character recognition: A comprehensive evaluation, *Pattern Recognition* 72 (2017) 1 – 14.
- [58] M. Budnik, E. L. Gutierrez-Gomez, B. Safadi, G. Qunot, Learned features versus engineered features for semantic video indexing, in: *13th International Workshop on Content-Based Multimedia Indexing (CBMI)*, 2015, pp. 1–6.