Multimed Tools Appl manuscript No.
(will be inserted by the editor)

Local Derivative Pattern for Action Recognition in Depth Images

Xuan Son Nguyen · Thanh Phuong Nguyen · François Charpillet · Ngoc-Son Vu

Received: date / Accepted: date

Abstract This paper proposes a new local descriptor for action recognition in depth images using second-order directional Local Derivative Patterns (LDPs). LDP relies on local derivative direction variations to capture local patterns contained in an image region. Our proposed local descriptor combines different directional LDPs computed from three depth maps obtained by representing depth sequences in three orthogonal views and is able to jointly encode the shape and motion cues. Moreover, we suggest the use of Sparse Coding-based Fisher Vector (SCFVC) for encoding local descriptors into a global representation of depth sequences. SCFVC has been proven effective for object recognition but has not gained much attention for action recognition. We perform action recognition using Extreme Learning Machine (ELM). Experimental results on three public benchmark datasets show the effectiveness of the proposed approach.

Keywords Action recognition · Local Derivative Pattern · Sparse Coding · Fisher Vector · Extreme Learning Machine

Xuan Son Nguyen
Université de Caen Basse-Normandie, CNRS, GREYC, UMR 6072, 14000 Caen, France
E-mail: xuan-son.nguyen@unicaen.fr

Thanh Phuong Nguyen
Aix Marseille Université, CNRS, ENSAM, LSIS, UMR 7296, 13397 Marseille, France
Université de Toulon, CNRS, LSIS, UMR 7296, 83957 La Garde, France
E-mail: thanh-phuong.nguyen@univ-tln.fr

François Charpillet
INRIA 54600 Villers-Lès-Nancy, France
CNRS, LORIA, UMR 7503, 54600 Villers-Lès-Nancy, France
E-mail: francois.charpillet@inria.fr

Ngoc-Son Vu
ETIS UMR 8051, Université Paris Seine, UCP, ENSEA, CNRS, 95000 Cergy, France
E-mail: son.vu@ensea.fr

1 Introduction

Human action recognition is an important topic of computer vision with many applications, such as assisted living, smart surveillance, sports video analysis and health monitoring. Traditional approaches mainly focused on recognizing action in RGB images. However, these approaches have limitations due to the sensitivity to lighting change and the lack of structure information of RGB images. Since the introduction of affordable 3D depth sensing cameras, many approaches for human action recognition in depth images have been proposed and achieved impressive results. Unlike RGB images, depth images provide 3D information of the scene which greatly reduces depth ambiguity. Depth sensors are also insensitive to lighting change, which enables human action recognition under varying lighting conditions. Existing approaches for action recognition in depth images can be broadly grouped into three main categories: skeleton-based, depth map-based and hybrid approaches. Yang and Tian [40] learned EigenJoints from differences of joint positions and used Naïve-Bayes-Nearest-Neighbor [2] for action classification. Vemulapalli et al. [35] used rotations and translations to represent 3D geometric relationships of body parts in a Lie group [25], and then employed Dynamic Time Warping [24] and Fourier Temporal Pyramid [38] to model the temporal dynamics. Evangelidis et al. [8] proposed skeletal quad which describes the positions of nearby joints in the human skeleton and used Fisher Vector (FV) [31] for feature encoding. Wang et al. [36] represented actions by histograms of spatial-part-sets and temporal-part-sets, where spatial-part-sets are sets of frequently co-occurring spatial configurations of body parts in a single frame, and temporal-part-sets are co-occurring sequences of evolving body parts. This approach has been shown to be robust to ambiguous poses. Luo et al. [21] proposed a dictionary learning approach where group sparsity and geometry constraint were incorporated to increase the discriminative power. A temporal pyramid matching scheme was used to keep the temporal information in action descriptors. Du et al. [7] divided the human skeleton into five parts, and fed them to five subnets of a recurrent neural network [32]. The representations extracted by the subnets at a layer were hierarchically fused to be the inputs of higher layers. Once the final representations of skeleton sequences have been obtained, actions were classified using a fully connected layer and a softmax layer. This approach requires low computational cost and can be used for online applications. While most of skeleton-based approaches produce low-dimensional action descriptors, their limitation is that they rely on skeleton tracking which is unreliable when depth images are noisy or occlusions are present. Moreover, in scenarios where there are interactions between human and objects, features extracted from 3D joint positions cannot capture all the discriminative information for effective action recognition.

Depth map-based approaches usually rely on low-level features from the space-time volume of depth sequences to compute action descriptors. Compared to skeleton-based ones, they do not require a skeleton tracker and thus can be used for more general scenarios. Li et al. [18] proposed a bag of 3D points to capture the shape of the human body and used an action graph [17] to model the dynamics of actions. In order to reduce the computational cost, only points on the contours of the projections of depth sequences on three orthogonal Cartesian planes were used. This approach has been shown to be robust to occlusion. Yang et al. [41] extracted HOG descriptors [23] from Depth Motion Maps (DMMs) obtained by projecting

depth sequences onto three orthogonal Cartesian planes. Chen et al. [5] proposed a real-time approach that used Local Binary Pattern (LBP) [27] computed for overlapped blocks in the DMMs of depth sequences to create action descriptors. Action recognition was performed using feature-level fusion and decision-level fusion. These approaches have limitation as the temporal order of shape and motion cues is ignored by projecting the whole sequence into one image for each plane. In order to address this issue, Liang et al. [19] proposed layered depth motion (LDM) feature which improved DMMs by computing the energy of the motion history at multilayered temporal intervals. In this way, the temporal order of shape and motion cues is also taken into account in LDM, although this temporal order is not fully captured. Kurakin et al. [15] proposed cell occupancy-based and silhouette-based features which were then used with action graphs for gesture recognition. This approach can give real-time performance. However, the motion cue is ignored in action descriptors and it is only applicable to hand gesture recognition. Wang et al. [37] introduced random occupancy patterns which were computed from sub-volumes of the space-time volume of depth sequences with different sizes and at different locations. This approach has the same limitation as that of [15] since the motion cue is not encoded into action descriptors. In order to overcome the limitations of the above approaches, Oreifej and Liu [28] and Yang and Tian [39] relied on surface normals in 4D space of depth, time, and spatial coordinates to jointly capture the shape and motion cues in local descriptors. These approaches and ours share a similar idea in that derivatives of depth values along the spatial and temporal dimensions are used to jointly encode the shape and motion cues. However, our approach exploits different directional derivatives to obtain a richer descriptor than these approaches. Moreover, we rely on binary patterns to encode the relationship of directional derivatives at the neighbourhood of each pixel. Thus, our proposed action descriptor is more informative while remaining relatively compact for efficient action recognition. Song et al. [33] constructed action descriptors from local surface patches extracted around trajectories of interest points in depth sequences. This approach requires RGB images to track interest points and thus is not applicable when only depth images are available. Rahmani and Mian [30] proposed a view-invariant approach by learning a deep convolutional neural network that represents different human body shapes and poses observed from numerous viewpoints in a view-invariant high-level space.

Hybrid approaches combine skeletal data and depth maps (or features which can be easily extracted from depth maps). Chaaraoui and Padilla-Lopez [4] combined normalized 3D joint positions and a radial silhouette-based feature to create action descriptors. This approach simply concatenates the joint-based and silhouette-based features to form the final action descriptor, which is an ad hoc solution. Zhu et al. [44] fused spatio-temporal features based on 3D interest point detectors and joint-based features using pair-wise joint distances in one frame and joints difference between two consecutive frames. This approach uses Random Forest [3] for feature fusion instead of using an ad hoc solution as the approach of [4]. However, it heavily depends on joint-based features as its performance drops when only spatio-temporal features are used. Wang et al. [38] introduced local occupancy patterns computed in spatio-temporal cells around 3D joint positions which were treated as the depth appearance of these joints. Since local features are extracted around 3D joint positions, this approach critically depends on skeleton tracking to construct action descriptors.

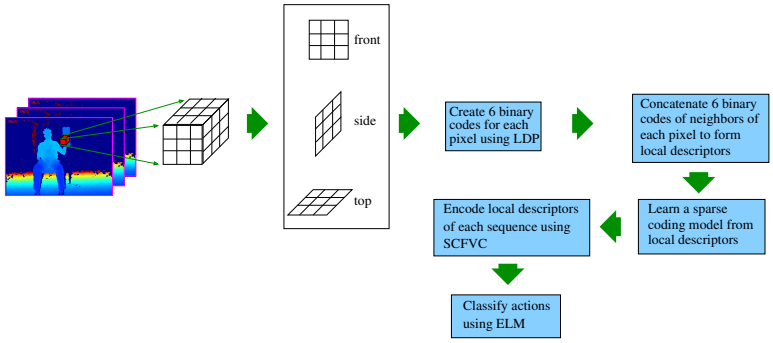

Fig. 1: The different steps of our method.

Our approach is closely related to the approaches of [43, 26]. These approaches rely on the concept of spatio-temporal slices to build descriptors for motion analysis [26] and facial expressions [43], which have a similar spirit as our approach. However, differently from these approaches, we do not build descriptors separately for horizontal and vertical slices. Instead, patterns calculated from the image plane and those calculated from horizontal and vertical slices are combined to form a local descriptor at each pixel in the space-time volume of a sequence. These local descriptors are then aggregated to obtain a global representation of the sequence. Thus, our approach can jointly capture the shape and motion cues for effective action recognition.

The paper is organized as follows. Section 2 gives an overview of our method. Section 3 explains our proposed local descriptor. Section 4 describes SCFVC for feature encoding. Section 5 presents ELM for action classification. In Section 6, we report the results of our experimental evaluation. Finally, Section 7 offers some conclusions and ideas for future work.

2 Overview of the proposed method

The main steps of our method are illustrated in Fig. 1. First, a set of 6 binary codes is calculated for each pixel in the space-time volume of a sequence using second-order directional Local Derivative Patterns [42]. This step relies on the representation of a sequence on the front, side and top views. The local descriptor for each pixel is formed by concatenating the 6 binary codes of that pixel and those of its neighbors, resulting in a 162-dimensional descriptor for each pixel. A sequence is now represented as a set of 162-dimensional vectors. Next, a sparse coding model is learned from the set of 162-dimensional vectors of the training sequences. The learned sparse coding model and the set of 162-dimensional vectors of each sequence are then used in a feature encoding scheme called Sparse Coding-based Fisher Vector [20] to obtain a global representation of that sequence. The global representations of all sequences are finally fed to a classifier based on Extreme Learning Machine [14] to obtain action labels. A detailed discussion of different components of the proposed method is given in the next sections.

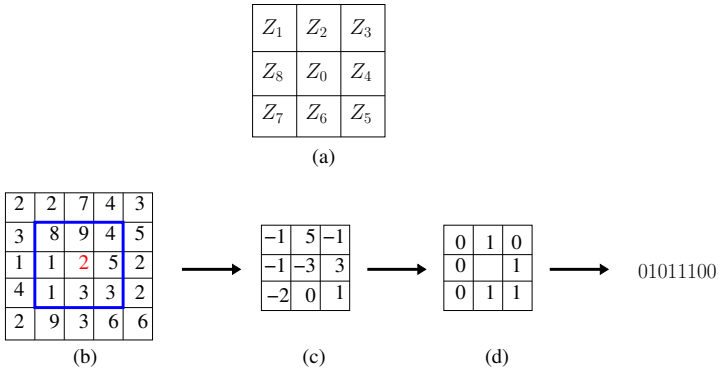

Fig. 2: (Best viewed in color) Example to calculate $LDP^2_\alpha(Z_0)$ with $\alpha = 0$. The referenced pixel Z_0 is marked in red.

3 Local Feature Descriptor

3.1 Local Derivative Pattern

Zhang et al. [42] proposed LDP which relies on local derivative direction variations to capture local patterns contained in an image region. Denote by I an intensity image, Z a general pixel, $I(Z)$ the intensity value of Z in I, $I'_\alpha(Z)$ where $\alpha = 0^0, 45^0, 90^0$ and 135^0 the first-order derivatives along $0^0, 45^0, 90^0$ and 135^0 directions respectively. Let Z_0 be a pixel in I, and Z_i, $i = 1, \ldots, 8$ be the neighboring pixel around Z_0 (see Fig. 2(a)). The four first-order derivatives at $Z = Z_0$ can be written as:

$$I'_{0^0}(Z_0) = I(Z_0) - I(Z_4) \tag{1}$$

$$I'_{45^0}(Z_0) = I(Z_0) - I(Z_3) \tag{2}$$

$$I'_{90^0}(Z_0) = I(Z_0) - I(Z_2) \tag{3}$$

$$I'_{135^0}(Z_0) = I(Z_0) - I(Z_1) \tag{4}$$

The second-order directional LDP, $LDP^2_\alpha(Z_0)$, in α direction at $Z = Z_0$ is defined as:

$$LDP^2_\alpha(Z_0) = \{f(I'_\alpha(Z_0), I'_\alpha(Z_1)), \ldots, f(I'_\alpha(Z_0), I'_\alpha(Z_8))\} \tag{5}$$

where $f(.,.)$ is a binary coding function which encodes the co-occurrence of two derivative directions at different neighboring pixels, defined as:

$$f(I'_\alpha(Z_0), I'_\alpha(Z_i)) = \begin{cases} 0, & \text{if } I'_\alpha(Z_i).I'_\alpha(Z_0) > 0 \\ 1, & \text{otherwise} \end{cases}$$

for $i = 1, \ldots, 8$.

An example to calculate $LDP^2_\alpha(Z_0)$ with $\alpha = 0$ is given in Fig. 2, where the 5×5 array shown in Fig. 2(b) represents an image patch and the number in each

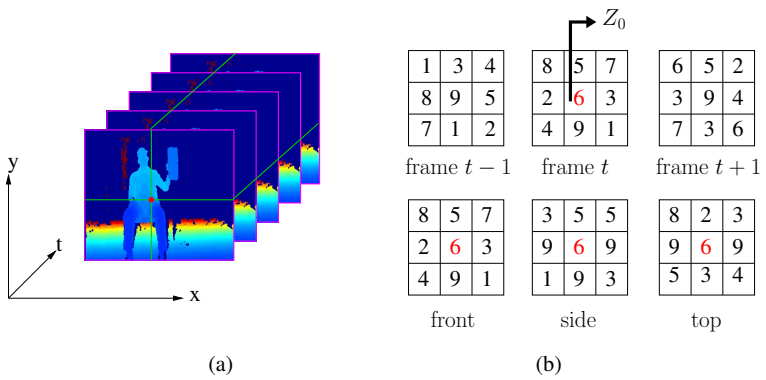

Fig. 3: (Best viewed in color) Example of (a) the front, side and top views of a sequence and (b) 8-neighborhood around a pixel for computing our proposed local descriptor.

cell represents the depth value of the corresponding pixel. The referenced pixel Z_0 is marked in red. The neighboring pixels of Z_0 are inside the 3×3 patch with blue edges. The first-order derivatives in direction $\alpha = 0$ at Z_0 and its neighboring pixels are calculated by subtracting the depth value of each pixel by that of its right neighbor, shown in Fig. 2(c). By multiplying the value of the central pixel in Fig. 2(c) and that of one of its neighbors and then encoding the obtained result with one bit (0 if the result is positive, 1 otherwise), we obtain 8 bits shown in Fig. 2(d). We now take the bits corresponding to the positions of $Z_1, Z_2, \ldots, Z_8$ to form a 8-bit binary number, which gives $LDP^2_\alpha(Z_0)$.

Note that the n^{th}-order directional LDPs for $n \geq 3$ can also be defined [42] by generalizing the above idea. In the following, we present the method for constructing our local descriptor using the second-order directional LDPs, but the same method can be applied for constructing local descriptors using the n^{th}-order directional LDPs for $n \geq 3$.

3.2 Our Proposed Local Descriptor

Our proposed local descriptor relies on the second-order directional LDPs calculated at each pixel of the depth sequence. In order to capture both the shape and motion cues, these LDPs are calculated using the neighborhood of each pixel from three depth maps obtained by representing the depth sequence in the front, side and top views respectively. Fig. 3(a) illustrates the construction of the three depth maps for a pixel. The referenced pixel is marked in red. The depth map of the front view corresponds to that of the current frame. The one of the side view corresponds to the image plane parallel to the y-t plane that passes through Z_0, which identifies one frame of the side view. The one of the top view corresponds to the image plane parallel to the x-t plane that passes through Z_0, which identifies one frame of the top view. Thus, for each sequence, the numbers of frames of the side and top views are equal to the width and height of a depth image. Fig. 3(b)

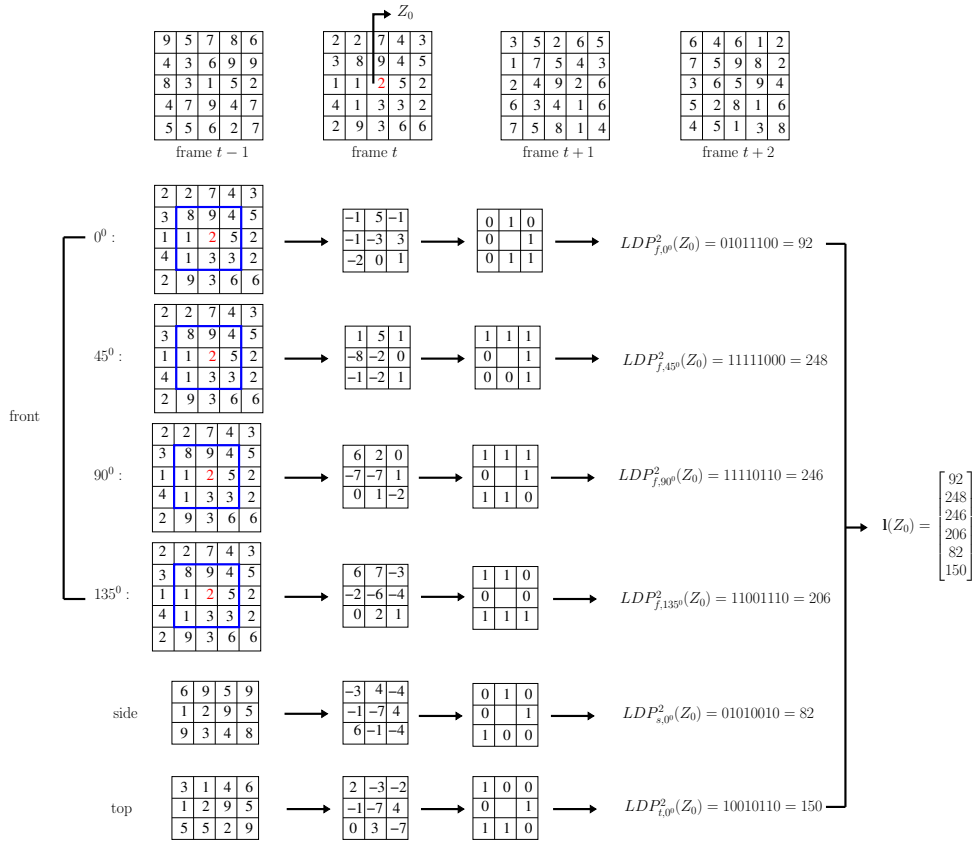

Fig. 4: (Best viewed in color) Example to calculate our proposed local descriptor.

shows an example of neighborhood used for calculating the directional LDPs in our local descriptor, where the three 3×3 patches are at the same coordinates in the image plane but are at three consecutive frames, and the numbers in each patch represent the depth values of the pixels. The referenced pixel Z_0 is at frame t and marked in red. The neighborhood of Z_0 in the front view are the 8 pixels around Z_0 at frame t. Those in the side view are obtained by taking the second columns from the three patches, and those in the top view are obtained by taking the second rows from the three patches.

In our proposed local descriptor, the shape cue is captured using the second-order directional LDPs calculated from the depth map of the front view, while the motion cue is introduced using those calculated from the depth maps of the side and top views. Formally, our local descriptor $\mathbf{l}(Z_0)$ can be written as follows:

$$\begin{aligned}\mathbf{l}(Z_0) = [&LDP^2_{f,0^0}(Z_0), LDP^2_{f,45^0}(Z_0), LDP^2_{f,90^0}(Z_0),\\ &LDP^2_{f,135^0}(Z_0), LDP^2_{s,0^0}(Z_0), LDP^2_{t,0^0}(Z_0)]^T.\end{aligned}$$

where $LDP^2_{f,0^0}(Z_0)$, $LDP^2_{f,45^0}(Z_0)$, $LDP^2_{f,90^0}(Z_0)$ and $LDP^2_{f,135^0}(Z_0)$ are the second-order directional LDPs obtained from the front view, $LDP^2_{s,0^0}(Z_0)$ and $LDP^2_{t,0^0}(Z_0)$ are those obtained from the side and top views respectively. Note that in our case, depth values are used in Eqs. 1, 2, 3, 4 instead of intensity values.

Fig. 4 shows the different steps to obtain our local descriptor. For the shape cue, four different binary patterns are calculated from the front view, which are 92, 248, 246 and 206 respectively. For the motion cue, two binary patterns are calculated from the side and top views, which are 82 and 150 respectively. The six binary patterns are then combined to obtain a 6-dimensional vector that captures both the local shape and motion cues at Z_0.

In order to keep the correlation between directional LDPs of neighboring pixels, we concatenate the 6-dimensional vectors from a local spatio-temporal neighborhood of Z_0. In the example of Fig. 3(b), a 6-dimensional vector is calculated for each pixel of the three 3×3 patches at frames $t-1$, t, $t+1$ and then these vectors are concatenated to form the local descriptor at Z_0. Thus, our proposed local descriptors are 162-dimensional vectors.

4 Feature Encoding

Given a depth sequence which can be written as a set of whitened vectors $\mathbf{V} = \{\mathbf{v}_i, i = 1, \ldots, M\}$, where $\mathbf{v}_i \in \mathbb{R}^{162}$, M is the number of local descriptors extracted in the sequence, an encoding method is used to construct the global representation of the depth sequence. In the following, we explain SCFVC for this purpose.

FV relies on the assumption that $p(.;\theta)$ is a Gaussian mixture with a fixed number of components K. As the dimensionality of the feature space increases, K must also increase to model the feature space accurately. This results in the explosion of the FV representation dimensionality. In order to deal with this problem, Liu et al. [20] proposed SCFVC which assumes that local descriptors are drawn from a Gaussian distribution $\mathcal{N}(\mathbf{Bu}, \sigma I)$, where $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_K]$ is a matrix of bases (visual words) and $\mathbf{u}$ is a latent coding vector randomly generated from a zero mean Laplacian distribution. This corresponds to modeling $p(.;\theta)$ using a Gaussian mixture with an infinite number of components. The generative model can be written as:

$$p(\mathbf{v}, \mathbf{u}|\mathbf{B}) = p(\mathbf{v}|\mathbf{u}, \mathbf{B})p(\mathbf{u})$$

Thus:

$$p(\mathbf{v}) = \int_{\mathbf{u}} p(\mathbf{v}, \mathbf{u}|\mathbf{B})d\mathbf{u} = \int_{\mathbf{u}} p(\mathbf{v}|\mathbf{u}, \mathbf{B})p(\mathbf{u})d\mathbf{u}$$

Denote $\mathbf{u}^* = \arg\max_{\mathbf{u}} p(\mathbf{v}|\mathbf{u}, \mathbf{B})p(\mathbf{u})$, then $p(\mathbf{v})$ can be approximated by:

$$p(\mathbf{v}) \approx p(\mathbf{v}|\mathbf{u}^*, \mathbf{B})p(\mathbf{u}^*)$$

Taking the logarithm of $p(\mathbf{v})$, one obtains:

$$\log(p(\mathbf{v}|\mathbf{B})) = \min_{\mathbf{u}} \frac{1}{\sigma^2}||\mathbf{v} - \mathbf{Bu}||_2^2 + \lambda||\mathbf{u}||_1. \quad (6)$$

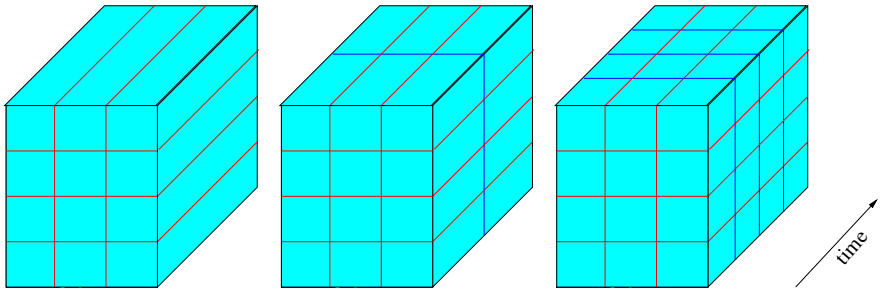

Fig. 5: The spatio-temporal grids used in our experiments. From left to right: the first, second and third temporal pyramids.

Eq. 6 reveals the relationship between the generative model and a sparse coding model. Liu et al. [20] showed that the FV representation of $\mathbf{V}$ is given by:

$$\mathbf{x} = [\mathbf{x}^1; \ldots; \mathbf{x}^K],$$
$$\mathbf{x}^k = \sum_{i=1}^{M} u_i^*(k)(\mathbf{v}_i - \mathbf{B}\mathbf{u}_i^*),$$

where $\mathbf{u}_i^*$ is the solution of the sparse coding problem, $u_i^*(k)$ is the k^{th} dimension of $\mathbf{u}_i^*$.

In order to capture the spatial geometry and temporal order of a depth sequence, we partition the space-time volume of the sequence into the spatio-temporal grids illustrated in Fig. 5, where the spatial grids are computed on the largest bounding box of the action, and the temporal grids are computed using the method of [39]. This method relies on the motion energy to partition the sequence into a set of temporal segments with different lengths instead of those with equal length as in the method of [16]. The motion energy characterizes the relative motion status at each frame with respect to the entire action and is calculated as follows:

$$e(t) = \sum_{v=1}^{3} \sum_{i=1}^{t-1} \text{sum}(|I_{i+1}^v - I_i^v| > \epsilon),$$

where I^1, I^2, I^3 are the depth maps obtained by projecting the sequence onto three orthogonal planes [41]; I_i^v is the i^{th} frame of I^v; $e(t)$ is the motion energy of frame t; ϵ is a threshold used for generating the binary map $|I_{i+1}^v - I_i^v| > \epsilon$; $\text{sum}(|I_{i+1}^v - I_i^v| > \epsilon)$ returns the number of non-zero elements in this map.

At the i^{th} pyramid level, the sequence is partitioned into a set of temporal segments $\{t_0t_1, t_1t_2, \ldots, t_{2^{i-1}-1}t_{2^{i-1}}\}$ such that $\bar{e}(t_j) = j/2^{i-1}$, $j = 0, \ldots, 2^{i-1}$, where $\bar{e}(t_j)$ is the normalized motion energy of frame t_j. This allows to pool local descriptors within temporal segments containing the same total amount of motion, which can deal better with the variations in motion speed and frequency when different people perform the same action than the method of [16]. We compute one FV for each grid, and concatenate the FVs of the grids to create the final representation of the sequence.

5 Extreme Learning Machine

ELM was originally introduced for the single-hidden-layer feedforward neural networks [14] and then extended to the generalized SLFNs [11,12]. Given a set of training data $(\mathbf{x}_i, \mathbf{q}_i)$, $i = 1, \ldots, N$ where $\mathbf{x}_i \in \mathbf{R}^d$ and $\mathbf{q}_i = [q_{i,1}, \ldots, q_{i,C}]^T \in \mathbf{R}^C$ is the class label so that if the original class label of $\mathbf{x}_i$ is l, then only the l^{th} element of $\mathbf{q}_i$ is one, the remaining elements are zero. Denote by L the number of hidden nodes, $\mathbf{h}(\mathbf{x}) = [h_1(\mathbf{x}), \ldots, h_L(\mathbf{x})]$ the output vector of the hidden layer with respect to the input $\mathbf{x}$, $\boldsymbol{\beta} = [\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_C]$, $\boldsymbol{\beta}_j \in \mathbf{R}^L$ the vector of the output weights linking hidden layer to the j^{th} output node. The output function of ELM is given as:

$$\mathbf{f}(\mathbf{x}) = \mathbf{h}(\mathbf{x})\boldsymbol{\beta}. \tag{7}$$

$\mathbf{h}(\mathbf{x})$ can be seen as a feature mapping since it maps the data from a d-dimensional space to a L-dimensional space. In ELM, the model is learned by solving the following optimization problem:

$$\min_{\boldsymbol{\beta},\boldsymbol{\xi}} \frac{1}{2}||\boldsymbol{\beta}||^2 + \zeta \frac{1}{2}\sum_{i=1}^{N} ||\boldsymbol{\xi}_i||^2,$$

$$\text{subject to: } \mathbf{h}(\mathbf{x}_i)\boldsymbol{\beta} = \mathbf{q}_i^T - \boldsymbol{\xi}_i^T,\ i = 1, \ldots, N,$$

where ζ is a user-specified parameter and provides a tradeoff between the distance of the separating margin and the training error, $\boldsymbol{\xi}_i = [\xi_{i,1}, \ldots, \xi_{i,C}]^T$ is the training error vector of the C output nodes with respect to the training sample $\mathbf{x}_i$.

Assuming that the number of training samples is not huge, the solution of the above problem is given as [13]:

$$\boldsymbol{\beta} = \mathbf{H}^T \Big(\frac{\mathbf{I}}{\zeta} + \mathbf{H}\mathbf{H}^T\Big)^{-1}\mathbf{Q}, \tag{8}$$

where $\mathbf{Q} = [\mathbf{q}_1^T; \ldots; \mathbf{q}_N^T]$ and $\mathbf{H} = [\mathbf{h}(\mathbf{x}_1); \ldots; \mathbf{h}(\mathbf{x}_N)]$ is the hidden-layer output matrix. From Eqs. 7 and 8, the output function of ELM classifier is given by:

$$\mathbf{f}(\mathbf{x}) = \mathbf{h}(\mathbf{x})\boldsymbol{\beta} = \mathbf{h}(\mathbf{x})\mathbf{H}^T \Big(\frac{\mathbf{I}}{\zeta} + \mathbf{H}\mathbf{H}^T\Big)^{-1}\mathbf{Q}.$$

Denote by $f_j(\mathbf{x})$ the output function of the j^{th} output node, i.e. $\mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}), \ldots, f_C(\mathbf{x})]$. Then the predicted class label of sample $\mathbf{x}$ is:

$$label(\mathbf{x}) = \arg \max_{j \in \{1,\ldots,C\}} f_j(\mathbf{x}).$$

If the feature mapping $\mathbf{h}(\mathbf{x})$ is unknown to users, one can define a kernel matrix for ELM as follows:

$$\boldsymbol{\Omega}_{ELM} = \mathbf{H}\mathbf{H}^T : \Omega_{ELMi,j} = h(\mathbf{x}_i).h(\mathbf{x}_j) = \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j).$$

In this case, the decision function of ELM classifier can be rewritten as:

$$\begin{aligned}\mathbf{f}(\mathbf{x}) &= \mathbf{h}(\mathbf{x})\mathbf{H}^T \Big(\frac{\mathbf{I}}{\zeta} + \mathbf{H}\mathbf{H}^T\Big)^{-1}\mathbf{Q} \\ &= \begin{bmatrix} \mathcal{K}(\mathbf{x}, \mathbf{x}_1) \\ \vdots \\ \mathcal{K}(\mathbf{x}, \mathbf{x}_N) \end{bmatrix}^T \Big(\frac{\mathbf{I}}{\zeta} + \Omega_{ELM}\Big)^{-1}\mathbf{Q}.\end{aligned}$$

6 Experiments

In this section, we evaluate the proposed method on three benchmark datasets: MSRAction3D [18], MSRActionPairs3D [28] and MSRGesture3D [37]. We also compare it against several state-of-the-art methods to demonstrate its effectiveness. In our experiments, the sparse coding model was learned using the SPAMS library [22]. In order to analyze the impact of different components of our method on its performance, and to demonstrate the effectiveness of the proposed local descriptor, we implemented the five methods **Ours_3rdOrder**, **Ours_4thOrder**, **Ours_FV**, **Ours_SVM**, **LBP-TOP** [43]. **Ours_3rdOrder** and **Ours_4thOrder** were obtained by replacing the second-order directional LDPs in our method with the third-order and the fourth-order ones, respectively. Note that we do not show the results obtained using the n^{th}-order directional LDPs with $n > 4$ as the performance of our method on the three datasets dropped when n reached to 4. **Ours_FV** was obtained by replacing SCFVC in the feature encoding step of our method with FV. We used the VLFeat library [34] to compute the FVs. **Ours_SVM** was obtained by replacing ELM in the classification step of our method with SVM. We used LIBLINEAR [9] as the linear SVM classifier. For **Ours_FV**, the number of components in the Gaussian mixture model was set to 50, which was experimentally found to give the best results. For **LBP-TOP**, three different histograms corresponding to the front, side and top views were built for each sequence. The binary codes computed on all frames of one view were used to build the histogram corresponding to that view. The three histograms were then concatenated to obtain the descriptor of a sequence. In order to capture the spatial geometry and temporal order of a sequence, we used the same spatio-temporal grids as our method (see Fig. 5). The final descriptor of a sequence is the concatenation of the descriptors of the grids. Note that in the original paper, **LBP-TOP** descriptors were computed using $2 \times 2 \times 1$ spatio-temporal grids created by 2×2 spatial grids and one temporal pyramid level. However, in our experiments, we observed that **LBP-TOP** achieved better accuracy with our spatio-temporal pyramid representation. For action recognition, **LBP-TOP** used the same ELM-based classifier as our method.

6.1 Parameter Settings

In this section, we investigate the performance of our method with respect to the number of visual words K and the number of temporal pyramids. Fig. 6 shows the recognition accuracies of our method when $K = 60, 80, 100, 120, 140$. As can be observed, our method gives the best results on the three datasets with $K = 100$. Good results are also obtained with $K = 140$. Fig. 7 shows the recognition accuracies of our method when one, two and three temporal pyramids are used. When the number of temporal pyramids changes, our method gives the best results with 3 temporal pyramids on MSRGesture3D and MSRActionPairs3D, and with 2 or 3 temporal pyramids on MSRAction3D. Since our method achieves the best results on the three datasets with $K = 100$ and 3 temporal pyramids, in the following we report the results obtained using this setting.

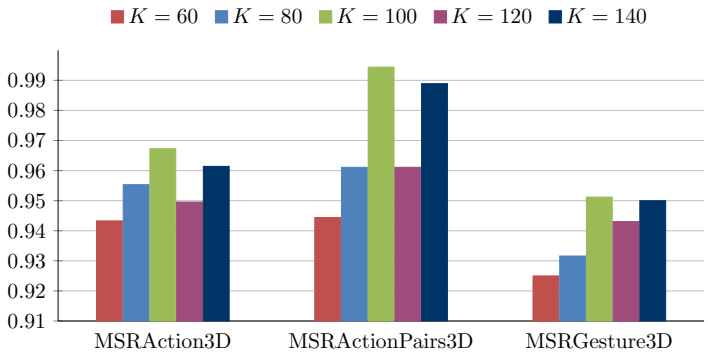

Fig. 6: (Best viewed in color) Accuracy w.r.t the number of visual words K.

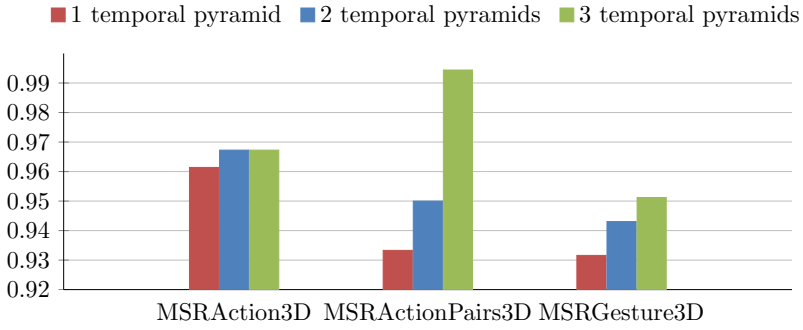

Fig. 7: (Best viewed in color) Accuracy w.r.t the number of temporal pyramids.

6.2 MSRAction3D Dataset

The MSRAction3D is an action dataset captured using a depth sensor similar to Kinect. It contains 20 actions performed by 10 different subjects. Each subject performs every action two or three times.

Our experimental setting was based on the discussion of Padilla-López et al. [29]. Specifically, we used the same experimental setting proposed in [18] as it was widely adopted by previous approaches, where the 20 actions were divided into three subsets AS1, AS2 and AS3, each having 8 actions. The AS1 and AS2 were intended to group actions with similar movements, while AS3 was intended to group complex actions together. Action recognition was performed on each subset separately. We used the most challenging cross-subject test, where subjects 1,3,5,7,9 were used for training and subjects 2,4,6,8,10 were used for testing. In Tab. 1, we compare our method against other state-of-the-art methods. As can be observed, our method achieves the highest accuracy for AS1. The average accuracy over the three subsets is 96.72%, which is the second best result among

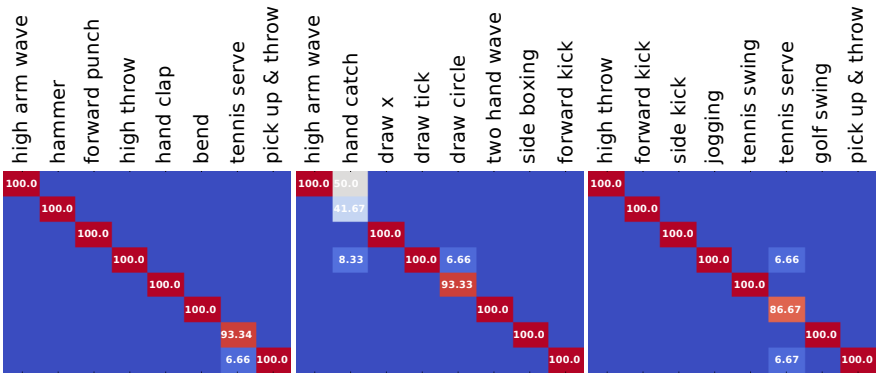

Fig. 8: (Best viewed in color) The confusion matrices of our method on MSRAction3D (left: AS1, middle: AS2, right: AS3).

Method	AS1	AS2	AS3	Ave.
Chen et al., 2013 [6]	96.2	83.2	92.0	90.47
Gowayyed et al., 2013 [10]	92.39	90.18	91.43	91.26
Zhu et al., 2013 [44]	-	-	-	94.3
Luo et al., 2013 [21]	97.2	**95.5**	**99.1**	**97.27**
Vemulapalli et al., 2014 [35]	95.29	83.87	98.21	92.46
Chen et al., 2015 [5]	98.1	92.0	94.6	94.9
Du et al., 2015 [7]	93.33	94.64	95.50	94.49
Liang et al., 2016 [19]	97.2	92.9	94.6	94.9
Ours	**99.05**	92.92	98.21	96.72
Ours_3rdOrder	97.17	**95.58**	97.32	96.69
Ours_4thOrder	96.23	94.69	96.43	95.78
Ours_FV	90.57	87.61	93.75	90.64
Ours_SVM	96.23	85.84	97.32	93.13
LBP-TOP	87.74	74.34	87.5	83.19

Table 1: Recognition accuracy comparison of our method and previous methods on AS1, AS2, AS3 of MSRAction3D.

the competing ones. The method of [21] performs slightly better than our method. However, this method relies on skeleton tracking to obtain 3D joint positions which is unreliable when depth images are noisy or severe occlusions are present. Note that our method outperforms the method of [5] which uses LBP for constructing local descriptors and the method of [7] which relies on a recurrent neural network. When one temporal pyramid is used, our method gives an average accuracy of 96.14%, demonstrating that the directional LDPs computed from the side and top views capture well the motion cue for accurate action recognition. **Ours_3rdOrder** gives a similar accuracy as our method, while **Ours_4thOrder**

Method	Accuracy
Yang et al., 2012 [41]	66.11
Wang et al., 2013 [38]	82.22
Oreifej and Liu, 2013 [28]	96.67
Yang and Tian, 2014 [39]	98.89
Amor et al., 2016 [1]	93.00
Rahmani and Mian, 2016 [30]	**99.44**
Ours	**99.44**
Ours_3rdOrder	95.00
Ours_4thOrder	91.11
Ours_FV	97.78
Ours_SVM	97.22
LBP-TOP	87.78

Table 2: Recognition accuracy comparison of our method and previous methods on MSRActionPairs3D.

has an accuracy of 95.78%, 0.94% inferior to our method. These results show that the second-order directional LDPs perform similarly or better than the third-order and the fourth-order ones on MSRAction3D. Our method outperforms **Ours_FV** by 8.48% on AS1, 5.31% on AS2 and 4.46% on AS3, showing that SCFVC is better than FV in terms of accuracy on the three subsets. By using ELM instead of SVM for action classification, our method achieves better results on the three subsets and its average accuracy over the three subsets is increased by 3.59%. The training times per sequence of the ELM-based and SVM-based classifiers are approximately 63.64 milliseconds (ms) and 196 ms, respectively. The testing times per sequence of the ELM-based and SVM-based classifiers are approximately 0.04 ms and 29.31 ms, respectively. These results show that the ELM-based classifier is 3 times faster than the SVM-based one in the training phase, and it is 732 times faster than the SVM-based one in the testing phase. Thus, ELM is better than SVM not only in terms of accuracy but also in terms of computation time on MSRAction3D. Note that our method significantly outperforms **LBP-TOP**, demonstrating that it captures the joint shape-motion cues better than **LBP-TOP**. The confusion matrices of our method are shown in Fig. 8. Most of the confusions are between the actions *tennis serve* and *pick up and throw* (AS1), *high arm wave* and *hand catch* (AS2), *tennis serve* and *jogging* and *pick up and throw* (AS3).

6.3 MSRActionPairs3D Dataset

The MSRActionPairs3D is a paired-activity dataset of depth sequences captured by a depth camera. The dataset contains 6 pairs of activities, which were selected so that within each pair the motion and the shape cues are similar, but their correlations vary. There are 10 subjects with each subject performing each activity three times. This dataset is useful to evaluate how well the descriptors capture the shape and motion cues jointly in the sequence.

We used the experimental setting described in [28], where the first five actors were used for testing, and the rest for training. Tab. 2 shows the accuracies of our method and different state-of-the-art methods on MSRActionPairs3D. Our method achieves an accuracy of 99.44%, which is the same accuracy as the recent work [30] based on a deep convolutional neural network. The accuracies of **Ours_3rdOrder**, **Ours_4thOrder**, **Ours_FV**, **Ours_SVM** and **LBP-TOP** are also given in Tab. 2. As can be observed, the performance of our method degrades when the order of local pattern is increased from the second-order directional LDPs to the third-order and the fourth-order ones, showing that the n^{th}-order directional LDPs with $n \geq 3$ do not improve the performance of our method on MSRActionPairs3D. When FV is used instead of SCFVC for feature encoding, the accuracy goes down to 97.78%, demonstrating that SCFVC is better than FV in terms of accuracy on MSRActionPairs3D. The accuracy of our method is 2.22% better than that of **Ours_SVM**. The training times per sequence of the ELM-based and SVM-based classifiers are approximately 98.53 ms and 233.61 ms, respectively. The testing times per sequence of the ELM-based and SVM-based classifiers are approximately 0.039 ms and 29.56 ms, respectively. Again, these results indicate that ELM is better than SVM in terms of both accuracy and computation time on MSRActionPairs3D. We can also observe that our method is significantly more accurate than **LBP-TOP**, which confirms the effectiveness of our proposed local descriptor compared to **LBP-TOP**. Since our method gives a high recognition accuracy, we do not show the confusion matrix for this dataset.

6.4 MSRGesture3D Dataset

The MSRGesture3D dataset contains 12 dynamic hand gestures defined by the American sign language. Each gesture is performed two or three times by 10 subjects.

We used the leave-one-subject-out cross validation scheme proposed by [37]. The accuracy of our method and different state-of-the-art methods is given in Tab. 3. Our method gives an accuracy of 95.12% which outperforms the competing ones. Experimental results reveal that the second-order directional LDPs perform the best over the third-order and the fourth-order ones on this dataset. Consistent with the results obtained in our previous experiments, SCFVC outperforms FV and the ELM-based classifier is more accurate than the SVM-based one. The training times per sequence of the ELM-based and SVM-based classifiers are approximately 208.41 ms and 314.47 ms, respectively. The testing times per sequence of the ELM-based and SVM-based classifiers are approximately 0.04 ms and 29.83 ms, respectively. These results again illustrate the advantages of ELM compared to SVM in our method. The accuracy of our method is 10.26% better than that of **LBP-TOP**, demonstrating its superiority in recognition accuracy over **LBP-TOP**. The confusion matrix is shown in Fig. 9. Most of the confusions are between the actions *store* and *finish*, *past* and *where*, *finish* and *bathroom*.

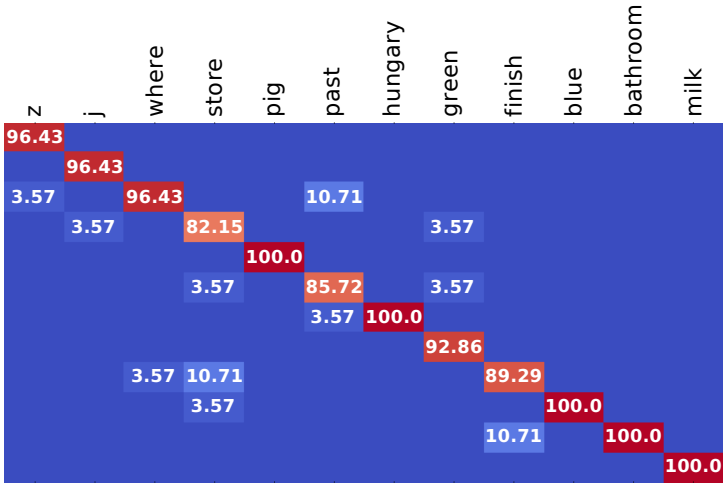

Fig. 9: (Best viewed in color) The confusion matrix of our method on MSRGesture3D.

Method	Accuracy
Kurakin et al., 2012 [15]	87.70
Wang et al., 2012 [37]	88.50
Yang et al., 2012 [41]	89.20
Oreifej and Liu, 2013 [28]	92.45
Yang and Tian, 2014 [39]	94.74
Rahmani and Mian, 2016 [30]	94.70
Liang et al., 2016 [19]	94.70
Ours	**95.12**
Ours_3rdOrder	94.03
Ours_4thOrder	92.22
Ours_FV	94.55
Ours_SVM	93.75
LBP-TOP	84.86

Table 3: Recognition accuracy comparison of our method and previous methods on MSRGesture3D.

7 Conclusions

We have proposed a new descriptor for action recognition in depth images. Our proposed descriptor encodes jointly the shape and motion cues using second-order directional LDPs. We have suggested SCFVC to effectively encode local descriptors into a global representation of depth sequences. Action recognition has been performed using ELM. We have presented the experimental evaluation on three benchmark datasets showing the effectiveness of the proposed method.

For future research, we study the fusion of the proposed descriptor with other descriptors based on depth and skeletal data in order to increase its accuracy.

References

1. Amor, B.B., Su, J., Srivastava, A.: Action Recognition Using Rate-Invariant Analysis of Skeletal Shape Trajectories. TPAMI **38**(1), 1–13 (2016)
2. Boiman, O., Shechtman, E., Irani, M.: In Defense of Nearest-Neighbor Based Image Classification. In: CVPR, pp. 1–8 (2008)
3. Breiman, L.: Random Forests. Machine Learning **45**(1), 5–32 (2001)
4. Chaaraoui, A.A., Padilla-Lopez, J.R., Florez-Revuelta, F.: Fusion of Skeletal and Silhouette-Based Features for Human Action Recognition with RGB-D Devices. In: ICCVW, pp. 91–97 (2013)
5. Chen, C., Jafari, R., Kehtarnavaz, N.: Action Recognition from Depth Sequences Using Depth Motion Maps-based Local Binary Patterns. In: WACV, pp. 1092–1099 (2015)
6. Chen, C., Liu, K., Kehtarnavaz, N.: Real-time Human Action Recognition Based on Depth Motion Maps. Journal of Real-Time Image Processing **12**(1), 155–163 (2016)
7. Du, Y., Wang, W., Wang, L.: Hierarchical Recurrent Neural Network for Skeleton Based Action Recognition. In: CVPR, pp. 1110–1118 (2015)
8. Evangelidis, G., Singh, G., Horaud, R.: Skeletal Quads: Human Action Recognition Using Joint Quadruples. In: ICPR, pp. 4513–4518 (2014)
9. Fan, R.E., Chang, K.W., Hsieh, C.J., Wang, X.R., Lin, C.J.: LIBLINEAR: A Library for Large Linear Classification. Journal of Machine Learning Research **9**, 1871–1874 (2008)
10. Gowayyed, M.A., Torki, M., Hussein, M.E., El-Saban, M.: Histogram of Oriented Displacements (HOD): Describing Trajectories of Human Joints for Action Recognition. In: IJCAI, pp. 1351–1357 (2013)
11. Huang, G.B., Chen, L.: Convex Incremental Extreme Learning Machine. Neurocomputing **70**(16-18), 3056–3062 (2007)
12. Huang, G.B., Chen, L.: Enhanced Random Search Based Incremental Extreme Learning Machine. Neurocomputing **71**(16-18), 3460–3468 (2008)
13. Huang, G.B., Zhou, H., Ding, X., Zhang, R.: Extreme Learning Machine for Regression and Multiclass Classification. IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics **42**(2), 513–529 (2012)
14. Huang, G.B., Zhu, Q.Y., Siew, C.K.: Extreme Learning Machine: A New Learning Scheme of Feedforward Neural Networks. In: IEEE International Joint Conference on Neural Networks, vol. 2, pp. 985–990 (2004)
15. Kurakin, A., Zhang, Z., Liu, Z.: A Read-Time System for Dynamic Hand Gesture Recognition with A Depth Sensor. In: EUSIPCO, pp. 1975–1979 (2012)
16. Laptev, I., Marszalek, M., Schmid, C., Rozenfeld, B.: Learning Realistic Human Actions from Movies. In: CVPR, pp. 1–8 (2008)
17. Li, W., Zhang, Z., Liu, Z.: Expandable Data-Driven Graphical Modeling of Human Actions Based on Salient Postures. IEEE Transactions on Circuits and Systems for Video Technology **18**(11), 1499–1510 (2008)
18. Li, W., Zhang, Z., Liu, Z.: Action Recognition Based on A Bag of 3D Points. In: CVPRW, pp. 9–14 (2010)
19. Liang, C., Chen, E., Qi, L., Guan, L.: Improving Action Recognition Using Collaborative Representation of Local Depth Map Feature. IEEE Signal Processing Letters **23**(9), 1241–1245 (2016)
20. Liu, L., Shen, C., Wang, L., van den Hengel, A., Wang, C.: Encoding High Dimensional Local Features by Sparse Coding Based Fisher Vectors. In: NIPS, pp. 1143–1151 (2014)
21. Luo, J., Wang, W., Qi, H.: Group Sparsity and Geometry Constrained Dictionary Learning for Action Recognition from Depth Maps. In: ICCV, pp. 1809–1816 (2013)
22. Mairal, J., Bach, F., Ponce, J., Sapiro, G., Jenatton, R., Obozinski, G.: SPAMS: SPArse Modeling Software, v2.4. URL http://spams-devel.gforge.inria.fr/downloads.html
23. Mikolajczyk, K., Schmid, C.: A Performance Evaluation of Local Descriptors. TPAMI **27**(10), 1615–1630 (2005)
24. Müller, M.: Information Retrieval for Music and Motion. Springer-Verlag New York, Inc. (2007)

25. Murray, R.M., Sastry, S.S., Zexiang, L.: A Mathematical Introduction to Robotic Manipulation. CRC Press, Inc. (1994)
26. Ngo, C.W., Pong, T.C., Chin, R.T.: Detection of Gradual Transitions through Temporal Slice Analysis. In: CVPR, pp. 36–41 (1999)
27. Ojala, T., Pietikainen, M., Harwood, D.: Performance Evaluation of Texture Measures with Classification Based on Kullback Discrimination of Distributions. In: Proceedings of the 12th IAPR International Conference on Pattern Recognition, vol. 1, pp. 582–585 (1994)
28. Oreifej, O., Liu, Z.: HON4D: Histogram of Oriented 4D Normals for Activity Recognition from Depth Sequences. In: CVPR, pp. 716–723 (2013)
29. Padilla-López, J.R., Chaaraoui, A.A., Flórez-Revuelta, F.: A Discussion on the Validation Tests Employed to Compare Human Action Recognition Methods Using the MSR Action3D Dataset. CoRR **abs/1407.7390** (2014)
30. Rahmani, H., Mian, A.: 3D Action Recognition from Novel Viewpoints. In: CVPR, pp. 1506–1515 (2016)
31. Sanchez, J., Perronnin, F., Mensink, T., Verbeek, J.: Image Classification with the Fisher Vector: Theory and Practice. IJCV **105**(3), 222–245 (2013)
32. Schmidhuber, J.: Deep Learning in Neural Networks: An Overview. Neural Networks **61**, 85–117 (2015)
33. Song, Y., Liu, S., Tang, J.: Describing Trajectory of Surface Patch for Human Action Recognition on RGB and Depth Videos. IEEE Signal Processing Letters **22**(4), 426–429 (2015)
34. Vedaldi, A., Fulkerson, B.: Vlfeat: An Open and Portable Library of Computer Vision Algorithms. In: Proceedings of the 18th ACM International Conference on Multimedia, pp. 1469–1472 (2010)
35. Vemulapalli, R., Arrate, F., Chellappa, R.: Human Action Recognition by Representing 3D Skeletons as Points in a Lie Group. In: CVPR, pp. 588–595 (2014)
36. Wang, C., Wang, Y., Yuille, A.L.: An Approach to Pose-Based Action Recognition. In: CVPR, pp. 915–922 (2013)
37. Wang, J., Liu, Z., Chorowski, J., Chen, Z., Wu, Y.: Robust 3D Action Recognition with Random Occupancy Patterns. In: ECCV, pp. 872–885 (2012)
38. Wang, J., Liu, Z., Wu, Y., Yuan, J.: Mining Actionlet Ensemble for Action Recognition with Depth Cameras. In: CVPR, pp. 1290–1297 (2012)
39. Yang, X., Tian, Y.: Super Normal Vector for Activity Recognition Using Depth Sequences. In: CVPR, pp. 804–811 (2014)
40. Yang, X., Tian, Y.L.: EigenJoints-based Action Recognition Using Naive-Bayes-Nearest-Neighbor. In: CVPRW, pp. 14–19 (2012)
41. Yang, X., Zhang, C., Tian, Y.: Recognizing Actions Using Depth Motion Maps-based Histograms of Oriented Gradients. In: Proceedings of the 20th ACM International Conference on Multimedia, pp. 1057–1060 (2012)
42. Zhang, B., Gao, Y., Zhao, S., Liu, J.: Local Derivative Pattern Versus Local Binary Pattern: Face Recognition With High-Order Local Pattern Descriptor. IEEE Transactions on Image Processing **19**(2), 533–544 (2010)
43. Zhao, G., Pietikainen, M.: Dynamic Texture Recognition Using Local Binary Patterns with an Application to Facial Expressions. TPAMI **29**(6), 915–928 (2007)
44. Zhu, Y., Chen, W., Guo, G.: Fusing Spatiotemporal Features and Joints for 3D Action Recognition. In: CVPRW, pp. 486–491 (2013)