Action Recognition from Depth Sequences Using Depth Motion Maps-based Local Binary Patterns

Chen Chen, Roozbeh Jafari, Nasser Kehtarnavaz
Department of Electrical Engineering
The University of Texas at Dallas
chenchen870713@gmail.com, {rjafari, kehtar}@utdallas.edu

Abstract

This paper presents a computationally efficient method for action recognition from depth video sequences. It employs the so called depth motion maps (DMMs) from three projection views (front, side and top) to capture motion cues and uses local binary patterns (LBPs) to gain a compact feature representation. Two types of fusion consisting of feature-level fusion and decision-level fusion are considered. In the feature-level fusion, LBP features from three DMMs are merged before classification while in the decision-level fusion, a soft decision-fusion rule is used to combine the classification outcomes. The introduced method is evaluated on two standard datasets and is also compared with the existing methods. The results indicate that it outperforms the existing methods and is able to process depth video sequences in real-time.

1. Introduction

Human action recognition has a wide range of human computer interaction applications including assisted living, smart surveillance, and gaming. It is also part of fitness training and health monitoring, *e.g.,* [6, 5]. Research on human action recognition initially involved the use of video sequences captured by traditional RGB video cameras, *e.g.,* [11, 20]. Space-time based methods such as local spatio-temporal features are popular techniques for video representation and have been shown promising performance in action recognition [20, 18]. However, intensity-based video images are sensitive to lighting conditions and background clutter which limit the robustness of action recognition.

Since the introduction of cost-effective depth cameras (*e.g.,* Microsoft Kinect), more recent research works on human action recognition have been carried out using depth images captured by such cameras, *e.g.,* [12, 28, 7, 13]. Compared with video images, depth images generated by a structured light or time-of-flight depth camera are insensitive to changes in lighting conditions. Depth images also provide 3D structural information to help distinguishing different poses. Moreover, human skeleton information can be estimated from depth images [19]. For example, the Kinect Windows SDK [1] is able to provide the estimated 3D positions and rotation angles of the body joints, which can be utilized as additional information to enhance the performance of action recognition.

In this paper, we present a computationally efficient action recognition framework using depth motion maps (DMMs)-based local binary patterns (LBPs) [14] and kernel-based extreme learning machine (KELM) [8]. For a depth video sequence, all the depth frames in the video are first projected onto three orthogonal Cartesian planes to form the projected images corresponding to three projection views [front (f), side (s), and top (t) views] . The absolute difference between two consecutive projected images is accumulated through the entire depth video creating three DMMs (DMM_f, DMM_s, and DMM_t) from the three projection views [7]. The DMMs are divided into overlapped blocks and the LBP operator is applied to each block to calculate an LBP histogram. The resulted LBP histograms of the blocks in a DMM are represented as a feature vector. Since there are three DMMs for a depth video sequence, both feature-level fusion and decision-level fusion approaches are investigated using KELM. Specifically, for the feature-level fusion, the LBP feature vector for each DMM is concatenated or stacked as a single feature vector before it is fed into a KELM classifier. Decision-level fusion operates on probability outputs of each classifier and combines individual decisions into a joint one. Here, three classifiers that use the LBP features generated from the three DMMs are considered and KELM is utilized to provide the probability outputs of each classification. A soft decision fusion scheme, logarithmic opinion pool (LOGP) rule [2], is employed to merge the probability outputs and to assign the final class label. The introduced action recognition pipeline is illustrated in Figure 1.

The contributions made in this paper are three-fold:

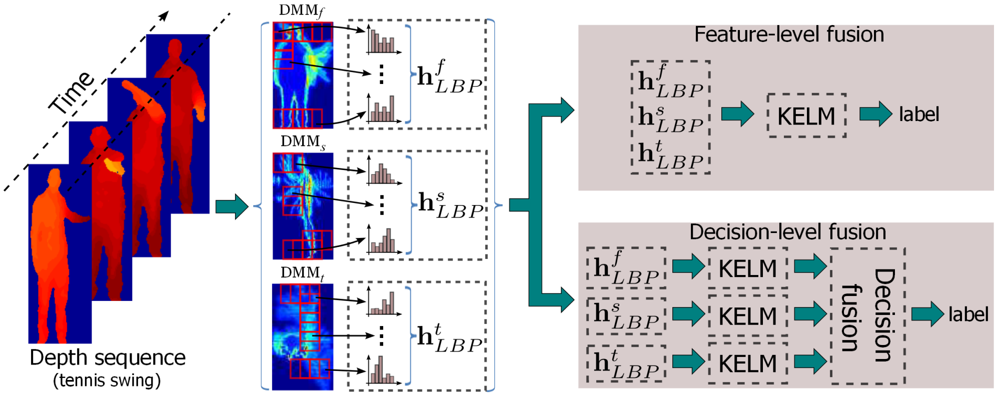

Figure 1. Pipeline of the developed action recognition method.

1. A computationally efficient and effective feature descriptor based on DMMs and LBPs is proposed. DMMs are used to capture motion cues in a depth video sequence. Then, the LBP operator, which is an effective measure of local image texture, is applied to the overlapped blocks within each DMM to represent it in a compact way in order to enhance the discriminatory power for action recognition.

2. Two levels of fusion (*i.e.,* feature-level fusion and decision-level fusion) are applied to the extracted LBP features from the three DMMs. Feature-level fusion involves feature concatenation of multiple features before classification, while decision-level fusion involves merging the probability outputs of each classification via the soft decision-fusion rule LOGP.

3. The developed action recognition pipeline is computationally efficient allowing it to run in real-time.

This paper is organized as follows. In Section 2, a brief review of previous works is presented. In Section 3, the details of the DMMs-based LBP features and the fusion approaches are described. The results are reported in Section 4. The conclusion appears in Section 5.

2. Previous Works

Many algorithms for action recognition from depth video sequences are based on low-level features extracted from depth images. In [12], the expandable graphical model was employed to model the temporal dynamics of actions and a bag of 3D points was used to model the postures. In [28], depth images were projected onto three orthogonal planes and accumulated to generate DMMs. The histograms of the oriented gradients (HOG) computed from DMMs were used as feature descriptors. The concept of DMMs was also considered in [7] with some modifications and a collaborative representation classifier was developed for action recognition. In [26], depth cuboid similarity features (DCSF) were built around the local spatio-temporal interest points (STIPs) extracted from depth video sequences to describe local 3D depth cuboids. In [24], random occupancy pattern (ROP) features were extracted from depth video sequences and a sparse coding approach was utilized to encode these features. The results demonstrated robustness to occlusion. In [15], a histogram of oriented 4D surface normals (HON4D) was constructed to capture complex joint shape-motion cues at pixel-level.

Skeleton based approaches utilize the high-level skeleton information extracted from depth video sequences. In [27], skeleton joint locations were placed into 3D spatial bins to build histograms of 3D joint locations (HOJ3D) as features for action recognition. In [3], an evolutionary algorithm was used to select the optimal subset of skeleton joints based on the topological structure of a skeleton leading to improved recognition rates. In [25], a new feature called local occupancy pattern feature was used for action recognition. An actionlet ensemble model was learnt to capture intra-class variations and to deal with noises and errors in depth images and joint positions. In [4], human actions were modeled by a spatio-temporal hierarchy of bio-inspired 3D skeletal features. Linear dynamical systems were employed to learn the dynamics of these features. In [22], a body part-based skeleton representation was proposed to model the relative geometry between body parts. Then, human actions were modeled as curves using a Lie group. Although some of the skeleton-based approaches achieved high recognition performance, skeleton-based methods are not applicable for applications where skeleton information is not available.

3. Introduced Recognition Method

3.1. DMMs calculation

DMMs were initially introduced in [28]. The concept of DMMs was also considered in [7] where the procedure for generating DMMs was modified to reduce the computational complexity. In this paper, we adopt the method of generating DMMs described in [7] due to its computational efficiency. Specifically, given a depth video sequence with N frames, each frame in the video is projected onto three orthogonal Cartesian planes to form three 2D projected maps, denoted by map_f, map_s, and map_t. DMMs are then generated as follows:

$$DMM_{\{f,s,t\}} = \sum_{j=1}^{N-1} |map^{j+1}_{\{f,s,t\}} - map^{j}_{\{f,s,t\}}|, \qquad (1)$$

where j is the frame index. A bounding box is set to extract the non-zero region (region of interest) in each DMM. Here, the foreground extracted DMMs are considered as the final DMMs. An example of the three DMMs for a *tennis swing* depth sequence is shown in Figure 1. The motion characteristics can be effectively captured by DMMs.

3.2. DMMs-based LBP features

The LBP operator [14] is a simple yet effective gray scale and rotation invariant texture operator that has been used in various applications. It labels pixels in an image with decimal numbers that encode local texture information. Given a pixel (scalar value) g_c in an image, its neighbor set contains pixels that are equally spaced on a circle of radius r ($r > 0$) with the center at g_c. If the coordinates of g_c are $(0, 0)$ and m neighbors $\{g_i\}_{i=0}^{m-1}$ are considered, the coordinates of g_i are $(-rsin(2\pi i/m), rcos(2\pi i/m))$. The gray values of circular neighbors that do not fall in the image grids are estimated by bilinear interpolation [14]. Figure 2 illustrates an example of a neighbor set for $(m = 8, r = 1)$ (the values for m and r may change in practice). The LBP is created by thresholding the neighbors $\{g_i\}_{i=0}^{m-1}$ with the center pixel g_c to generate an m-bit binary number. The resulting LBP for g_c can be expressed in decimal form as follows:

$$LBP_{m,r}(g_c) = \sum_{i=0}^{m-1} U(g_i - g_c)2^i, \qquad (2)$$

where $U(g_i - g_c) = 1$ if $g_i \geq g_c$ and $U(g_i - g_c) = 0$ if $g_i < g_c$. Although the LBP operator in Eq.(2) produces 2^m different binary patterns, a subset of these patterns, named uniform patterns, is able to describe image texture [14]. After obtaining the LBP codes for pixels in an image, an occurrence histogram is computed over an image or a region to represent the texture information.

In Figure 3, the LBP-coded image corresponding to the DMM_f of a *two hand wave* depth sequence is shown. It can be observed that the edges in the LBP-coded image are more enhanced compared with the DMM_f. Therefore, the DMMs are first generated for a depth sequence, then the LBP operator is applied to overlapped blocks of the DMMs. The LBP histograms of the blocks for each DMM are concatenated to form the feature vector denoted by $\mathbf{h}^f_{LBP}$, $\mathbf{h}^s_{LBP}$, and $\mathbf{h}^t_{LBP}$. The feature extraction procedure is illustrated in Figure 1. The resulted DMMs-based LBP features provide a compact representation of the DMMs.

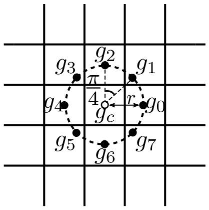

Figure 2. Center pixel g_c and its 8 circular neighbors $\{g_i\}_{i=0}^{7}$ with radius $r = 1$.

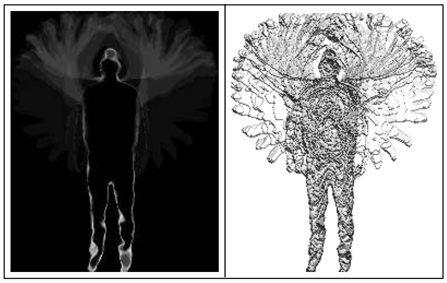

Figure 3. LBP-coded image (right) corresponding to the DMM_f (left) of a *two hand wave* depth sequence. The pixel values of the LBP-coded image are LBPs in decimal form.

3.3. KELM Classification

Extreme learning machine (ELM) was developed for single-hidden-layer feed-forward neural networks (SLFNs) [9]. Unlike traditional feed-forward neural networks that require all the parameters to be tuned, the hidden node parameters in ELM are randomly generated leading to a much faster learning rate.

For C classes, let us define class labels to be $y_k \in \{0, 1\}(1 \leq k \leq C)$. Thus, a row vector $\mathbf{y} = [y_1, \ldots, y_k, \ldots, y_C]$ indicates the class to which a sample belongs. For example, if $y_k = 1$, then the sample belongs to the kth class. Given n training samples $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{n}$, where $\mathbf{x}_i \in \mathbb{R}^M$ and $\mathbf{y}_i \in \mathbb{R}^C$, the model of a single hidden layer neural network having L hidden nodes can be expressed as

$$\sum_{j=1}^{L} \boldsymbol{\beta}_j h(\mathbf{w}_j \cdot \mathbf{x}_i + e_j) = \mathbf{y}_i, \quad i = 1, ..., n, \qquad (3)$$

where $h(\cdot)$ is a nonlinear activation function (*e.g.,* Sigmoid function), $\boldsymbol{\beta}_j \in \mathbb{R}^C$ denotes the weight vector connecting the jth hidden node to the output nodes, $\mathbf{w}_j \in \mathbb{R}^M$ denotes the weight vector connecting the jth hidden node to the input nodes, and e_j is the bias of the jth hidden node. The term $\mathbf{w}_j \cdot \mathbf{x}_i$ denotes the inner product of $\mathbf{w}_j$ and $\mathbf{x}_i$. For n

equations, (3) can be written in this compact form

$$\mathbf{H}\boldsymbol{\beta} = \mathbf{Y}, \tag{4}$$

where $\boldsymbol{\beta} = [\boldsymbol{\beta}_1^T ... \boldsymbol{\beta}_n^T]^T \in \mathbb{R}^{L\times C}$, $\mathbf{Y} = [\mathbf{y}_1^T ... \mathbf{y}_n^T]^T \in \mathbb{R}^{n\times C}$, and $\mathbf{H}$ is the hidden layer output matrix of the neural network expressed as

$$\mathbf{H} = \begin{bmatrix} \mathbf{h}(\mathbf{x}_1) \\ \vdots \\ \mathbf{h}(\mathbf{x}_n) \end{bmatrix} = \begin{bmatrix} h(\mathbf{w}_1 \cdot \mathbf{x}_1 + e_1) & \cdots & h(\mathbf{w}_L \cdot \mathbf{x}_1 + e_L) \\ \vdots & \ddots & \vdots \\ h(\mathbf{w}_1 \cdot \mathbf{x}_n + e_1) & \cdots & h(\mathbf{w}_L \cdot \mathbf{x}_n + e_L) \end{bmatrix}. \tag{5}$$

$\mathbf{h}(\mathbf{x}_i) = [h(\mathbf{w}_1 \cdot \mathbf{x}_i + e_1), ..., h(\mathbf{w}_L \cdot \mathbf{x}_i + e_L)]$ is the output of the hidden nodes in response to the input $\mathbf{x}_i$. Since in most cases, $L \ll n$, the smallest norm least-squares solution of (4) described in [9] can be used. That is

$$\boldsymbol{\beta}' = \mathbf{H}^\dagger \mathbf{Y}, \tag{6}$$

where $\mathbf{H}^\dagger$ is the Moore-Penrose generalized inverse of matrix $\mathbf{H}$, $\mathbf{H}^\dagger = \mathbf{H}^T(\mathbf{H}\mathbf{H}^T)^{-1}$. A positive value $\frac{1}{\rho}$ is normally added to the diagonal elements of $\mathbf{H}\mathbf{H}^T$ as a regularization term. As a result, the output function of the ELM classifier can be expressed as

$$\mathbf{f}_L(\mathbf{x}_i) = \mathbf{h}(\mathbf{x}_i)\boldsymbol{\beta} = \mathbf{h}(\mathbf{x}_i)\mathbf{H}^T \left(\frac{\mathbf{I}}{\rho} + \mathbf{H}\mathbf{H}^T \right)^{-1} \mathbf{Y}. \tag{7}$$

If the feature mapping $\mathbf{h}(\mathbf{x}_i)$ is unknown, a kernel matrix for ELM can be considered as follows:

$$\Omega_{ELM} = \mathbf{H}\mathbf{H}^T : \Omega_{ELM_{i,j}} = \mathbf{h}(\mathbf{x}_i) \cdot \mathbf{h}(\mathbf{x}_j) = K(\mathbf{x}_i, \mathbf{x}_j). \tag{8}$$

Hence, the output function of KELM is given by

$$\mathbf{f}_L(\mathbf{x}_i) = \begin{bmatrix} K(\mathbf{x}_i, \mathbf{x}_1) \\ \vdots \\ K(\mathbf{x}_i, \mathbf{x}_n) \end{bmatrix}^T \left(\frac{\mathbf{I}}{\rho} + \Omega_{ELM} \right)^{-1} \mathbf{Y}. \tag{9}$$

The label of a test sample $\mathbf{x}_l$ is assigned to the index of the output node with the largest value, *i.e.,*

$$y_l = \underset{k=1,...,C}{\arg\max}\, f_L(\mathbf{x}_l)_k, \tag{10}$$

where $f_L(\mathbf{x}_l)_k$ denotes the kth output of $\mathbf{f}_L(\mathbf{x}_l) = [f_L(\mathbf{x}_l)_1, \ldots, f_L(\mathbf{x}_l)_C]^T$. Compared with ELM, KELM provides a better generalization performance and is more stable.

3.4. Classification fusion

The common feature-level fusion is first considered. The LBP features from three DMMs, $\mathbf{h}^f_{LBP}$, $\mathbf{h}^s_{LBP}$, and $\mathbf{h}^t_{LBP}$, are simply stacked into a composite feature vector for classification. Note that $\mathbf{h}^f_{LBP}$, $\mathbf{h}^s_{LBP}$, and $\mathbf{h}^t_{LBP}$ are normalized to have the unit length before concatenation.

Although the feature-level fusion is straightforward, it has the disadvantages of incompatibility of multiple feature sets and large dimensionality. Thus, the decision-level fusion is also considered to merge the results from a classifier ensemble as shown in Figure 1. Specifically, $\mathbf{h}^f_{LBP}$, $\mathbf{h}^s_{LBP}$, and $\mathbf{h}^t_{LBP}$ are treated as three different features. Each of them is used as the input to a KELM classifier. The soft decision fusion rule LOGP is employed here to combine the outcomes from the three classifiers to produce the final result. Since the output function (*i.e.,* Eq. (9)) of KELM estimates the accuracy of the output label, the posterior probabilities are estimated using the decision function. As noted by Platt [16], the probability should be higher for a larger value of the decision function. Therefore, $\mathbf{f}_L(\mathbf{x})$ is scaled to $[0, 1]$ and Platt's empirical analysis using a sigmoid function is adopted to approximate the posterior probabilities,

$$p(y_k|\mathbf{x}) = \frac{1}{1 + \exp(Af_L(\mathbf{x})_k + B)}. \tag{11}$$

For simplicity, A and B parameters are set to $A = -1$ and $B = 0$.

In LOGP, the posterior probability $p_q(y_k|\mathbf{x})$ associated with each classifier is used to estimate a global membership function,

$$P(y_k|\mathbf{x}) = \prod_{q=1}^{Q} p_q(y_k|\mathbf{x})^{\alpha_q}, \tag{12}$$

or

$$\log P(y_k|\mathbf{x}) = \sum_{q=1}^{Q} \alpha_q p_q(y_k|\mathbf{x}), \tag{13}$$

where Q is the number of classifiers ($Q = 3$ in our case) and $\{\alpha_q\}_{q=1}^{Q}$ are uniformly distributed classifier weights. The final class label y^* is obtained according to

$$y^* = \underset{k=1,...,C}{\arg\max}\, P(y_k|\mathbf{x}). \tag{14}$$

.

4. Experimental Results

Our action recognition method was tested on two standard public domain datasets: MSRAction3D dataset [12] and MSRGesture3D dataset [10]. These datasets were captured by commercial depth cameras. Figure 4 shows sample depth images of different actions from these two datasets. Let us label our feature-level fusion approach as DMM-LBP-FF and decision-level fusion approach as DMM-LBP-DF. In our experiments, the radial basis function (RBF) kernel was employed in KELM. The MATLAB code used for our experiments is available on our website [1].

[1] https://sites.google.com/site/chenresearchsite/

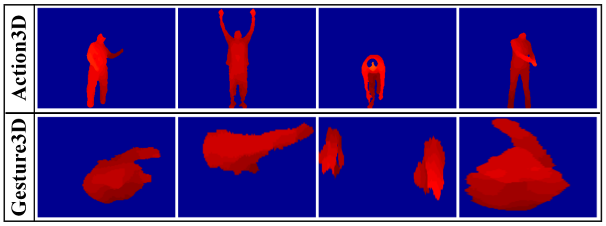

Figure 4. Sample depth images of different actions from the datasets MSRAction3D and MSRGesture3D.

4.1. MSRAction3D dataset

The MSRAction3D dataset [12] includes 20 actions performed by 10 subjects. The 20 actions are: *high wave*, *horizontal wave*, *hammer*, *hand catch*, *forward punch*, *high throw*, *draw x*, *draw tick*, *draw circle*, *hand clap*, *two hand wave*, *side boxing*, *bend*, *forward kick*, *side kick*, *jogging*, *tennis swing*, *tennis serve*, *golf swing*, and *pickup throw*. Each subject performed each action 2 or 3 times. This dataset includes 557 action sequences with 240 × 320 resolution and is a challenging dataset due to similarity of actions, *e.g., draw x* and *draw tick*. Two different experimental settings were used to test our method.

Experiment setting 1 - The same experimental setting reported in [12] was followed. Specifically, the actions were divided into three subsets ($AS1$, $AS2$ and $AS3$) as listed in Table 1. For each action subset (8 actions), three different tests were performed. In test one, 1/3 of the samples were used for training and the rest for testing; in test two, 2/3 of the samples were used for training and the rest for testing; in the cross subject test, one half of the subjects (1, 3, 5, 7, 9) were used for training and the rest for testing.

Experiment setting 2 - The same experimental setup in [24] was used. A total of 20 actions were employed and one half of the subjects (1, 3, 5, 7, 9) were used for training and the remaining subjects were used for testing. Setting 2 is considered more challenging than setting 1 because there are more action classes involved.

$AS1$	$AS2$	$AS3$
Horizontal wave	*High wave*	*High throw*
Hammer	*Hand catch*	*Forward kick*
Forward punch	*Draw x*	*Side kick*
High throw	*Draw tick*	*Jogging*
Hand clap	*Draw circle*	*Tennis swing*
Bend	*Two hand wave*	*Tennis serve*
Tennis serve	*Forward kick*	*Golf swing*
Pickup throw	*Side boxing*	*Pickup throw*

Table 1. Three action subsets of MSRAction3D dataset.

For our feature extraction, the DMMs of different action sequences were resized to have the same size for the purpose of reducing the intra-class variation. To have fixed sizes for DMM_f, DMM_s and DMM_t, the sizes of these maps for all the action samples in the dataset were found. The fixed size of each DMM was set to 1/2 of the mean value of all of the sizes. This made the sizes of DMM_f, DMM_s and DMM_t to be 102×54, 102×75 and 75×54, respectively. The block sizes of the DMMs were considered to be 25×27, 25×25 and 25×27 corresponding to DMM_f, DMM_s and DMM_t. The overlap between two blocks was taken to be one half of the block size. This resulted in 21 blocks for DMM_f, 35 blocks for DMM_s and 15 blocks for DMM_t.

Appropriate values for the parameter set (m, r) of the LBP features were also assigned. The feature-level fusion approach (DMM-LBP-FF) with setting 2 was considered for this purpose. The recognition results with various parameter sets are shown in Table 2. Note that the dimensionality of the LBP histogram feature based on uniform patterns is $m(m-1)+3$ [14], making the computational complexity of the LBP feature extraction dependent on the number of neighbors (m). Here, the LBP features for the depth sequences in the MSRAction3D dataset were calculated using $r = 1$ and various values of m. The average processing times associated with different parameter sets are shown in Figure 5. In our experiments, $m = 4$ and $r = 1$ were chosen in terms of recognition accuracy and computational complexity, making the dimensionalities of the LBP features $\mathbf{h}^f_{LBP}$, $\mathbf{h}^s_{LBP}$, and $\mathbf{h}^t_{LBP}$ 315, 525 and 225, respectively. To gain computational efficiency for the feature-level fusion, Principal Component Analysis (PCA) was applied to the concatenated feature vector to reduce the dimensionality. The PCA transform matrix was calculated using the features of the training data and the principal components that accounted for 95% of the total variation of the training features were considered. In all the experiments, the parameters for KELM (RBF kernel parameters) were chosen as the ones that maximized the training accuracy by means of a 5-fold cross-validation test.

MSRAction3D dataset						
r	1	2	3	4	5	6
$m = 4$	**91.94**	89.74	90.11	88.64	89.01	87.55
$m = 6$	**91.94**	90.11	90.48	90.11	89.01	90.11
$m = 8$	89.74	89.74	90.11	90.11	89.74	87.18
$m = 10$	**91.94**	90.11	90.11	89.74	88.64	88.28

Table 2. Recognition accuracy (%) of DMM-LBP-FF with different parameters (m, r) of LBP operator on MSRAction3D dataset (setting 2).

A comparison of our method with the existing methods was also carried out. The outcome of the comparison for setting 1 is listed in Table 3. As can be seen from this table, our method achieved superior recognition accuracy in most cases. In test two, our method even reached 100% recognition accuracy for all the three action subsets. In the

Method		Test one				Test two				Cross subject			
		$AS1$	$AS2$	$AS3$	Average	$AS1$	$AS2$	$AS3$	Average	$AS1$	$AS2$	$AS3$	Average
Li *et al.* [12]		89.5	89.0	96.3	91.6	93.4	92.9	96.3	94.2	72.9	71.9	79.2	74.7
DMM-HOG [28]		97.3	92.2	98.0	95.8	98.7	94.7	98.7	97.4	96.2	84.1	94.6	91.6
Chen *et al.* [7]		97.3	96.1	98.7	97.4	98.6	98.7	**100**	99.1	96.2	83.2	92.0	90.5
HOJ3D [27]		**98.5**	96.7	93.5	96.2	98.6	97.2	94.9	97.2	88.0	85.5	63.6	79.0
Chaaraoui *et al.* [3]		-	-	-	-	-	-	-	-	91.6	90.8	**97.3**	93.2
Vemulapalli *et al.* [22]		-	-	-	-	-	-	-	-	95.3	83.9	98.2	92.5
Space-Time Occupancy Patterns [23]		98.2	94.8	97.4	96.8	99.1	97.0	98.7	98.3	91.7	72.2	98.6	87.5
Ours	DMM-LBP-FF	96.7	**100**	**99.3**	**98.7**	**100**	**100**	**100**	**100**	98.1	92.0	94.6	**94.9**
	DMM-LBP-DF	98.0	97.4	**99.3**	98.2	**100**	**100**	**100**	**100**	**99.1**	**92.9**	92.0	94.7

Table 3. Comparison of recognition accuracies (%) of our method and other existing methods on MSRAction3D dataset using setting 1.

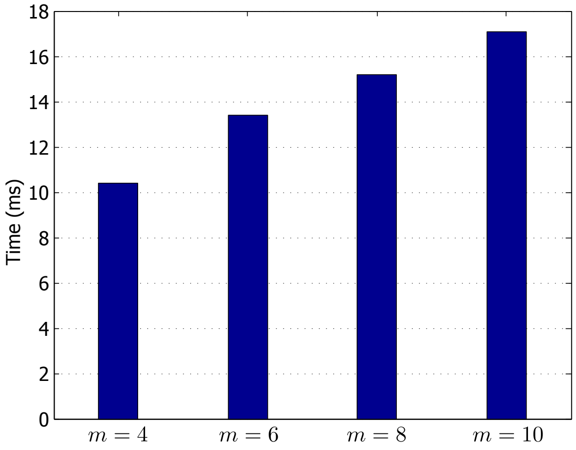

Figure 5. Average processing times (ms) of LBP feature extraction on MSRAction3D dataset with $r = 1$ and various values of m.

Method		Accuracy (%)
DMM-HOG [28]		85.5
Random Occupancy Patterns [24]		86.5
HON4D [15]		88.9
Actionlet Ensemble [25]		88.2
Depth Cuboid [26]		89.3
Tran *et al.* [21]		91.9
Rahmani *et al.* [17]		88.8
Vemulapalli *et al.* [22]		89.5
Ours	DMM-LBP-FF	91.9
	DMM-LBP-DF	**93.0**

Table 4. Comparison of recognition accuracy (%) on MSRAction3D dataset using setting 2.

cross subject test, our method also achieved superior performance.

The outcome using setting 2 is listed in Table 4. Again, it can be seen that our method outperformed the other methods. Although DMM-HOG [28] is a DMM-based method, the recognition accuracies of our method were more than 6% higher than that of DMM-HOG [28]. This indicated the DMM-based LBP features exhibit higher discriminatory power. This is because the overlapped block based feature extraction generates dense features, with LBP providing an effective representation of texture information. In addition, the recognition accuracy of the fusion-level fusion approach (DMM-LBP-FF) was found to be close to that of the decision-level fusion approach (DMM-LBP-DF). Figure 6 (a) shows the confusion matrix of DMM-LBP-DF for the MSRAction3D dataset, reflecting the overlaps among similar actions, for example, *hand catch* and *high throw*, and *draw x* and *draw tick*, due to the similarities of their DMMs.

To show that our method was not tuned to any specific training data, a cross validation experiment was conducted by considering all 252 combinations corresponding to choosing 5 subjects out 10 subjects for training and using the rest for testing. The results of our method (mean accuracy ± standard deviation (STD)) with the previously published results are provided in Table 5. Our method achieved superior performance compared with the other methods.

Method		Mean accuracy (%) ± STD
HON4D [15]		82.2 ± 4.2
Rahmani *et al.* [17]		82.7 ± 3.3
Tran *et al.* [21]		84.5 ± 3.8
Ours	DMM-LBP-FF	$\mathbf{87.9 \pm 2.9}$
	DMM-LBP-DF	87.3 ± 2.7

Table 5. Comparison of recognition accuracy (%) on MSRAction3D dataset using cross validation.

4.2. MSRGesture3D dataset

MSRGesture3D dataset [10] is a hand gesture dataset of depth sequences captured by a depth camera. This dataset contains a subset of gestures defined by American Sign Language (ASL). There are 12 gestures in the dataset: *bathroom*, *blue*, *finish*, *green*, *hungry*, *milk*, *past*, *pig*, *store*, *where*, *j*, *z*. The dataset contains 333 depth sequences, and is considered challenging because of self-occlusions. For this dataset, the leave-one-subject-out cross-validation test [24] was performed.

The fixed sizes for DMMs were determined using the same method for the MSRAction3D dataset. The sizes for DMM_f, DMM_s and DMM_t were 118×133, 118×29

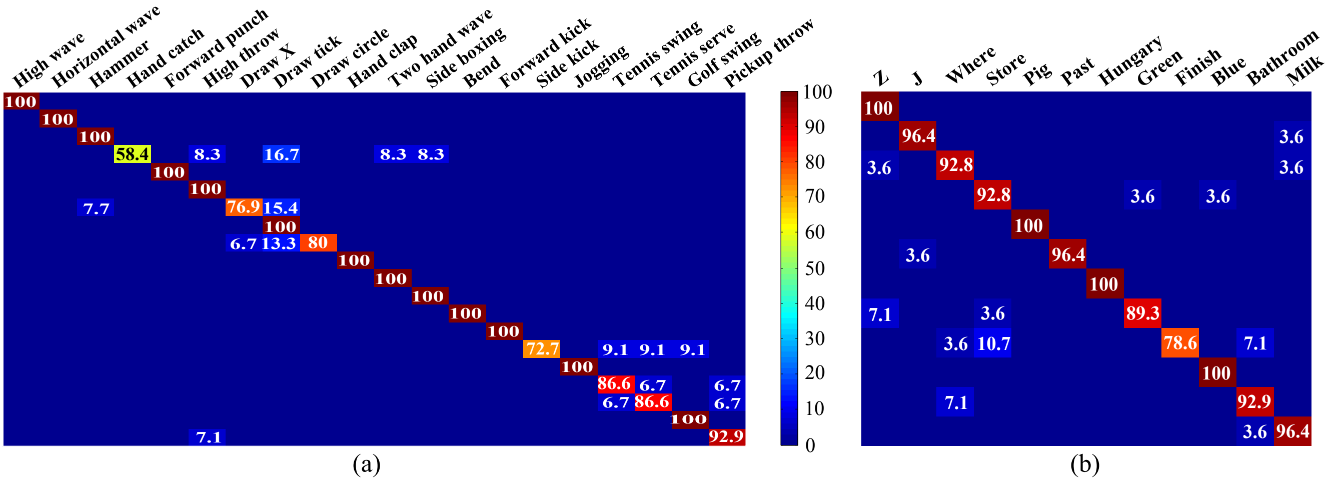

Figure 6. Confusion matrices of our decision-level fusion method (DMM-LBP-DF) for (a) MSRAction3D dataset (experiment setting 2) and (b) MSRGesture3D dataset.

and 29 × 133, respectively. The block sizes of the DMMs were 30 × 27, 30 × 15 and 15 × 27. $m = 10$ and $r = 1$ were used for LBP according to the recognition accuracies stated in Table 6.

MSRGesture3D dataset						
r	1	2	3	4	5	6
$m = 4$	92.49	92.49	91.89	91.29	91.89	91.89
$m = 6$	92.49	91.89	91.89	90.99	90.99	91.29
$m = 8$	91.59	90.99	90.69	91.89	91.29	90.99
$m = 10$	**93.39**	92.49	90.99	91.59	91.59	89.49

Table 6. Recognition accuracy (%) of DMM-LBP-FF with different parameters (m, r) of LBP operator on MSRGesture3D dataset.

Table 7 shows the recognition outcome of our method as well as seven existing methods. From Table 7, our decision-level fusion approach (DMM-LBP-DF) achieved the highest recognition accuracy. Note that three accuracies corresponding to three settings were reported in [17] and the accuracy 93.61% which was obtained using all 333 sequences is stated here. The confusion matrix of DMM-LBP-DF for MSRGesture3D dataset is shown in Figure 6 (b).

4.3. Computational efficiency

Finally, the computational efficiency or real-time aspect of our solution is presented here for the feature-level fusion approach as it is more computationally demanding. For a depth sequence, the DMMs calculation is performed frame by frame and the LBP features are extracted from the final DMMs. Therefore, there are four major processing components involved: DMMs calculation, LBP feature extraction, dimensionality reduction (PCA), and classification (KELM classifier). The average processing time of each component for the MSRAction3D dataset and the MSRGesture3D dataset is listed in Table 8. All experiments were carried out using MATLAB on an Intel i7 Quadcore 3.4 GHz desktop computer with 8GB of RAM. As noted in this table, our method met a real-time video processing rate of 30 frames per second.

Method		Accuracy (%)
Random Occupancy Patterns [24]		88.5
HON4D [15]		92.5
Rahmani *et al.* [17]		93.6
Tran *et al.* [21]		93.3
DMM-HOG [28]		89.2
Edge Enhanced DMM [29]		90.5
Kurakin *et al.* [10]		87.7
Ours	DMM-LBP-FF	93.4
	DMM-LBP-DF	**94.6**

Table 7. Comparison of recognition accuracy (%) on MSRGesture3D dataset.

Component	Time (ms)	
	MSRAction3D	MSRGesture3D
DMMs calculation	3.98±0.32/frame	4.77±1.42/frame
LBP feature extraction	10.39±1.08/sequence	19.20±0.83/sequence
PCA	0.36±0.05/sequence	0.52±0.04/sequence
Classification (KELM)	1.21±0.24/sequence	1.45±0.13/sequence

Table 8. Processing times (mean ± STD) associated with the components of our method.

5. Conclusion

In this paper, a computationally efficient and effective feature descriptor for action recognition was introduced. This feature descriptor was derived from depth motion maps (DMMs) and local binary patterns (LBPs) of a depth video sequence. DMMs were employed to capture the motion cues of actions, whereas LBP histogram features were used to achieve a compact representation of DMMs. Both fusion-level fusion and decision-level fusion approaches were considered which involved kernel-based extreme learning machine (KELM) classification. The experimental results on two standard datasets demonstrated improvements over the recognition performances of the existing methods.

References

[1] http://www.microsoft.com/en-us/kinectforwindows/. 1

[2] J. A. Benediktsson and J. R. Sveinsson. Multisource remote sensing data classification based on consensus and pruning. *IEEE Transactions on Geoscience and Remote Sensing*, 41(4):932–936, 2003. 1

[3] A. A. Chaaraoui, J. R. Padilla-López, P. Climent-Pérez, and F. Flórez-Revuelta. Evolutionary joint selection to improve human action recognition with rgb-d devices. *Expert Systems with Applications*, 41(3):786–794, 2014. 2, 6

[4] R. Chaudhry, F. Ofli, G. Kurillo, R. Bajcsy, and R. Vidal. Bio-inspired dynamic 3d discriminative skeletal features for human action recognition. In *CVPR Workshops*, pages 471–478, 2013. 2

[5] C. Chen, N. Kehtarnavaz, and R. Jafari. A medication adherence monitoring system for pill bottles based on a wearable inertial sensor. In *IEEE EMBS*, pages 4983–4986, 2014. 1

[6] C. Chen, K. Liu, R. Jafari, and N. Kehtarnavaz. Home-based senior fitness test measurement system using collaborative inertial and depth sensor. In *IEEE EMBS*, pages 4135–4138, 2014. 1

[7] C. Chen, K. Liu, and N. Kehtarnavaz. Real time human action recognition based on depth motion maps. *Journal of Real-Time Image Processing*, 2013. 1, 2, 6

[8] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang. Extreme learning machine for regression and multiclass classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 42(2):513–529, 2012. 1

[9] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew. Extreme learning machine: Theory and applications. *Neurocomputing*, 70(13):489–501, 2006. 3, 4

[10] A. Kurakin, Z. Zhang, and Z. Liu. A real time system for dynamic hand gesture recognition with a depth sensor. In *EUSIPCO*, pages 1975–1979, 2012. 4, 6, 7

[11] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *CVPR*, pages 1–8, 2008. 1

[12] W. Li, Z. Zhang, and Z. Liu. Action recognition based on a bag of 3d points. In *CVPR Workshops*, pages 9–14, 2010. 1, 2, 4, 5, 6

[13] K. Liu, C. Chen, R. Jafari, and N. Kehtarnavaz. Fusion of inertial and depth sensor data for robust hand gesture recognition. *IEEE Sensors Journal*, 14(6):1898–1903, 2014. 1

[14] T. Ojala, M. Pietikäinen, and T. Mäenpää. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971–987, 2002. 1, 3, 5

[15] O. Oreifej and Z. Liu. Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences. In *CVPR*, pages 716–723, 2013. 2, 6, 7

[16] J. Platt. Probabilistic outputs for support vector machines and comparison to regularized likelihood methods. In *Advances in Large Margin Classifiers*. MIT Press, Cambridge, MA, 1999. 4

[17] H. Rahmani, A. Mahmood, D. Q. Huynh, and A. Mian. Real time action recognition using histograms of depth gradients and random decision forests. In *WACV*, pages 626–633, 2014. 6, 7

[18] C. Schüldt, I. Laptev, and B. Caputo. Recognizing human actions: A local svm approach. In *ICPR*, pages 32–36, 2004. 1

[19] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *CVPR*, pages 1297–1304, 2011. 1

[20] J. Sun, X. Wu, S. Yan, L.-F. Cheong, T.-S. Chua, and J. Li. Hierarchical spatio-temporal context modeling for action recognition. In *CVPR*, pages 2004–2011, 2009. 1

[21] Q. D. Tran and N. Q. Ly. Sparse spatio-temporal representation of joint shape-motion cues for human action recognition in depth sequences. In *2013 IEEE RIVF International Conference on Computing and Communication Technologies, Research, Innovation, and Vision for the Future (RIVF)*, pages 253–258, 2013. 6, 7

[22] R. Vemulapalli, F. Arrate, and R. Chellappa. Human action recognition by representing 3d human skeletons as points in a lie group. In *CVPR*, 2014. 2, 6

[23] A. W. Vieira, E. R. Nascimento, G. L. Oliveira, Z. Liu, and M. F. Campos. On the improvement of human action recognition from depth map sequences using space-time occupancy patterns. *Pattern Recognition Letters*, 36:221–227, 2014. 6

[24] J. Wang, Z. Liu, J. Chorowski, Z. Chen, and Y. Wu. Robust 3d action recognition with random occupancy patterns. In *ECCV*, pages 872–885, 2012. 2, 5, 6, 7

[25] J. Wang, Z. Liu, Y. Wu, and J. Yuan. Mining actionlet ensemble for action recognition with depth cameras. In *CVPR*, pages 1290–1297, 2012. 2, 6

[26] L. Xia and J. Aggarwal. Spatio-temporal depth cuboid similarity feature for activity recognition using depth camera. In *CVPR*, pages 2834–2841, 2013. 2, 6

[27] L. Xia, C.-C. Chen, and J. Aggarwal. View invariant human action recognition using histograms of 3d joints. In *CVPR Workshops*, pages 20–27, 2012. 2, 6

[28] X. Yang, C. Zhang, and Y. Tian. Recognizing actions using depth motion maps-based histograms of oriented gradients. In *ACM Multimedia*, pages 1057–1060, 2012. 1, 2, 6, 7

[29] C. Zhang and Y. Tian. Edge enhanced depth motion map for dynamic hand gesture recognition. In *CVPR Workshops*, pages 500–505, 2013. 7