BoMW: Bag of Manifold Words for One-shot Learning Gesture Recognition from Kinect


Published in:
IEEE Transactions on Circuits and Systems for Video Technology

Document Version:
Peer reviewed version

Queen's University Belfast - Research Portal:
Link to publication record in Queen's University Belfast Research Portal

Publisher rights
© 2017 IEEE
This work is made available online in accordance with the publisher's policies. Please refer to any applicable terms of use of the publisher.

General rights
Copyright for the publications made accessible via the Queen's University Belfast Research Portal is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The Research Portal is Queen's institutional repository that provides access to Queen's research output. Every effort has been made to ensure that content in the Research Portal does not infringe any person's rights, or applicable UK laws. If you discover content in the Research Portal that you believe breaches copyright or violates any law, please contact openaccess@qub.ac.uk.
BoMW: Bag of Manifold Words for One-shot Learning Gesture Recognition from Kinect

Lei Zhang, Shengping Zhang, Feng Jiang, Yuankai Qi, Jun Zhang, Yuliang Guo, Huiyu Zhou

Abstract—In this paper, we study one-shot learning gesture recognition on RGB-D data recorded from Microsoft’s Kinect. To this end, we propose a novel bag of manifold words (BoMW) based feature representation on symmetric positive definite (SPD) manifolds. In particular, we use covariance matrices to extract local features from RGB-D data due to its compact representation ability as well as the convenience of fusing both RGB and depth information. Since covariance matrices are SPD matrices and the space spanned by them is the SPD manifold, traditional learning methods in the Euclidean space such as sparse coding can not be directly applied to them. To overcome this problem, we propose a unified framework to transfer the sparse coding on SPD manifolds to the one on the Euclidean space, which enables any existing learning method can be used. After building BoMW representation on a video from each gesture class, a nearest neighbor classifier is adopted to perform the one-shot learning gesture recognition. Experimental results on the ChaLearn gesture dataset demonstrate the outstanding performance of the proposed one-shot learning gesture recognition method compared against state-of-the-art methods. The effectiveness of the proposed feature extraction method is also validated on a new RGB-D action recognition dataset.

Index Terms—Gesture recognition, Covariance descriptor, Riemannian manifold, reproducing kernel Hilbert space, Kernel sparse coding.

I. INTRODUCTION

Human gestures provide a very useful way for our daily communication. For examples, when two normal persons are talking, gestures can be used to help one understand the other one better. For deaf people, gestures are their only way to communicate with other people. On the other hand, gestures can also be used for the interaction between human and computers. For example, people can use their hand gestures to control electrical devices at home. All these applications raise an interesting question that training a computer to automatically recognize human gestures will be very useful in different aspects including human computer interaction [1], [2], [3], [4], robot control [5], [6], [7], sign language recognition [8], [9], augmented reality [10] and so on. Due to its potential applications, gesture recognition has been attracting increasing attention in the computer vision community.

In the past few decades, a huge number of gesture recognition approaches have been proposed in the literature [11], [12], [13], [14], [15], [16], [17], [18], [19]. Although much effort has been devoted, the performance of gesture recognition is still not good enough for practical use because of two main challenges: 1) there are large variations in gesture movements. For example, when performing the same gesture, different people have different speeds, trajectories and spatial positions of the hands’ movement. Even when the same person performs a specific gesture at different times, the movements of the hands are also not identical; 2) it is very difficult to accurately track the hands. For example, when the hand and face are overlapped or the background is similar to skin color, tracking the hands may fail. These difficulties become more challenging for one-shot learning gesture recognition when only one sample per each class is given for training.

Recently, Microsoft’s Kinect has attracted increasing interest in both industry and research communities [20] since it can capture both RGB and depth information of a scene. When applied to gesture recognition, the depth information recorded by Kinect can be used to accurately track the human. Due to this appealing feature, it has been widely used in human action recognition [21] and also gesture recognition [22], [17]. For example, [22] first detects the hands using scene depth information and then employs Dynamic Time Warping for recognizing gestures. [17] extracts the static finger shape features from depth images and measures the dissimilarity between shape features for classification. In [23], a Bag of Features (BoF) model is first built upon the descriptors extracted at the detected points of interest from both RGB and depth images and then a nearest neighbour classifier is adopted for the classification. In [24], a multi-layered framework is proposed for gesture recognition, which first segment the moving hands and then extracts features from both the segmented semantic units and the whole gesture sequence. Although, the depth
information recorded by Kinect can help us to detect and track the moving hands, accurately segmenting the fingers is still very challenging since fingers have many complicated articulations and they usually occlude each other. Therefore, the performance of the existing methods that extracts features from the segmented hands can be degraded especially when the gestures are performed in a complicated background.

To overcome the drawbacks of the existing gesture recognition methods. In this paper, we propose a novel bag of manifold words (BoMW) based feature representation for gesture recognition. In contrast to most existing feature extraction methods for gesture recognition, the proposed BoMW does not depend on the accurate segmentation of the body or fingers and therefore is suitable for more practical scenes. In particular, we densely sample spatio-temporal cubics from both the RGB and depth videos. Then we extract motion features from RGB cubics and texture features from depth cubics. The motion and texture features from each cubic are then represented by a covariance descriptor. Since covariance matrices are symmetric positive definite (SPD) matrices and the space span by them is a SPD manifold, traditional learning methods in the Euclidean space such as sparse coding can not be directly applied to them. To overcome this problem, we propose to transfer the sparse coding on SPD manifolds to the one on the Euclidean space with the help of the Stein kernel, which maps SPD matrices into a vector space while preserving the geometry structures of the manifold. Subsequently, a bag of manifold words based feature representation is obtained for each video from each gesture class. Finally, a nearest neighbour classifier is adopted to perform the one-shot learning gesture recognition.

In summary, the contributions of the proposed method are three-folds:

- We propose to use covariance descriptors to extract motion and texture features from both RGB and depth videos. To our best knowledge, our work is the first to use covariance descriptors for RGB-D gesture recognition.
- We propose to transfer the sparse coding problem on SPD manifold to sparse coding on the Euclidean space via Stein kernel and then give the optimization solution.
- We carry out extensive experiments on the Chalearn gesture challenge dataset [25] and NTU RGB+D dataset [26] to demonstrate the effectiveness of the proposed algorithm with comparisons to the state-of-the-art methods. We also evaluate the performance of the proposed feature extraction method on a new RGB-D action recognition dataset.

The rest of the paper is organized as follows. Some related works are first reviewed in Section II. In Section III, we briefly review some basics of Hilbert spaces, CovDs and their geometry. The proposed BoMW method is introduced in Section IV. Experimental results are given in Section V. Secton VI concludes the paper.

II. RELATED WORK

The widely used features for gesture recognition are color [27], [28], shapes [29], [30] and motion [31], [32]. Compared to color and shape features, motion features extracted from two consecutive frames are more discriminative for gesture recognition because most of gestures can be distinguished by different motion patterns. To exploit motion information for gesture recognition, Agrawal and Chaudhuri [33] use the correspondences between patches in adjacent frames and then compute 2D motion histogram to represent the motion information. Shao and Ji [34] compute optical flow field from each frame and then use different combinations of the magnitudes and directions of optical flow field to compute a motion histogram. Zahedi et al. [35] combine skin color features and different first- and second-order derivative features to recognize sign language. Wong et al. [36] use PCA on motion gradient images of a sequence to obtain features for a Bayesian classifier. To extract motion features, Cooper et al. [37] extend haar-like features from spatial domain to spatio-temporal domain and proposes volumetric Haar-like features for gesture recognition.

Recently, depth information recorded from Kinect are used together with RGB data for gesture recognition. To extract more robust features from depth images for gesture recognition, Ren et al. [17] propose part based finger shape features, which do not depend on the accurate segmentation of the hands. Ming et al. [38] propose a new feature called 3D MoSIFT that is derived from MoSIFT [39]. Wan et al. [23] extend SIFT to spatio-temporal domain and propose 3D EMoSIFT and 3D SMoSIFT to extract features from RGB and depth images, which are invariant to scale and rotation, and have more compact and richer visual representations. Wan et al. [40] propose a discriminative dictionary learning method on 3D EMoSIFT features based on mutual information and then use sparse reconstruction for classification. Based on 3D Histogram of Flow (3DHOF) and Global Histogram of Oriented Gradient (GHOG), Fanello et al. [41] apply adaptive sparse coding to capture high-level feature patterns. Wu et al. [42] utilize both RGB and depth information and an extended-MHI representation is adopted as the motion descriptors. In [24], a multi-layered feature extraction method is proposed, which extracts features from both the segmented semantic units and the whole gesture sequence and then sequentially classifies the motion, location and shape components.

In the literature, many classifiers are used for gesture recognition, e.g., Dynamic Time Warping (DTW) [43], [44], [45], [24], linear SVMs [41], neuro-fuzzy inference system networks [46], hyper rectangular composite NNs [47], 3D Hopfield NN [48], sparse coding [49], [50], [51], [52], [53], [54]. Due to the ability of modeling temporal signals, Hidden Markov Model (HMM) is possibly the most well known classifier for gesture recognition. Bauer and Kraiss [55] propose to use HMM as classifier with 2D motion features for gesture recognition. Vogler [12] presents to use a parallel HMM for continuous gesture recognition. Fang et al. [56] propose a self-organizing feature maps/hidden Markov model (SOFM/HMM) for gesture recognition in which SOFM is used as an implicit feature extractor for continuous HMM. Recently, Wan et al. [57] propose SchMM to deal with the gesture recognition where sparse coding is adopted to find succinct representations and Lagrange dual is applied to learn the codebook.
One-shot learning gesture recognition is more challenging than traditional gesture recognition because only one training sample is available for each class. In the literature, several previous works have been focused on one-shot learning gesture recognition. For example, in [16], gesture sequences are viewed as third-order tensors and decomposed to three Stiefel manifolds and a natural metric is inherited from the factor manifolds. A geometric framework for least square regression is further presented and applied to gesture recognition. Mahbub et al. [32] propose a space-time descriptor and apply Motion History Imaging (MHI) techniques to track the motion flow in consecutive frames. Seo and Milanfar [58] present a novel action recognition method based on space-time locally adaptive regression kernels. Escalante et al. [59] introduce a novel action recognition method based on space-time locally adaptive regression kernels.

This problem attains its minimum value at \( x = \text{soft}(y, \lambda/2) \). The soft-threshold rule is the following non-linear function for \( a, T \in \mathbb{R} \):

\[
\text{soft}(a, T) = \begin{cases} 
  a + T & \text{if } a \leq -T, \\
  0 & \text{if } |a| \leq T, \\
  a - T & \text{if } a \geq T.
\end{cases}
\]

The minimization of \( j(x_i) \) is obtained by setting \( x_i \) to \( \text{soft}(y_i, \lambda/2) \). Because the variables in the function \( J(x) \) are uncoupled and the solution is obtained by minimizing with respect to each \( x_i \) individually, the minimizer of \( J(x) \) is obtained by applying soft-thresholding rule to each element, that is \( x = \text{soft}(y, \lambda/2) \).

**Definition 1.** Let \( F = [f_1 | f_2 | \cdots | f_m] \) be a \( d \times m \) matrix, obtained by stacking \( m \) independent observations \( f_i \in \mathbb{R}^d \) from an image (for example each observation may correspond to one pixel in an image). The covariance descriptor \( C \), as the name implies, is defined as

\[
C = \frac{1}{m-1} \sum_{i=1}^m (f_i - \mu)(f_i - \mu)^T,
\]

where \( \mu = \frac{1}{m} \sum_{i=1}^m f_i \) is the mean of the observations.

In this paper, we will use the following feature vector \( f_i \) as:

\[
f_j = \left( I_j, x, y, \frac{\partial I_j}{\partial x}, \frac{\partial I_j}{\partial y}, \frac{\partial^2 I_j}{\partial x^2}, \frac{\partial^2 I_j}{\partial y^2}, f_j^1, f_j^2, f_j^3 \right)^T,
\]

where \( I_j \) is the gray (color) value(s) at location \((x, y)\), and \( \partial/\partial x, \partial/\partial y, \partial^2/\partial x^2 \) and \( \partial^2/\partial y^2 \) are gradients and Laplacians along \( x \) and \( y \) dimensions, respectively. Apart from the widely used color, gradient and Laplacian features, we propose to use 8 extra textural features [62] for maintaining the tracking performance. In particular, \( f_j^1, f_j^2, f_j^3 \) and \( f_j^4, f_j^5, f_j^6 \) are the maximum responses across 6 orientations over 3 scales for two anisotropic filters, respectively. The remaining two features \( f_j^1 \) and \( f_j^6 \) are the responses of a Gaussian and a Laplacian of Gaussian filters both with \( \sigma = 10 \), respectively. Note that these 8-dimensional features can be extracted efficiently using a fast anisotropic Gaussian filter\(^1\) as shown in [62]. Fig. 1 shows the 8 textural responses of an example frame.

A CovD is a Symmetric Positive Definite (SPD) matrix with a well-known non-Euclidean structure. A \( d \times d \), real SPD matrix \( C \) has the property that \( \nu^T C \nu > 0 \) for all non-zero \( \nu \in \mathbb{R}^d \). The space of \( d \times d \) SPD matrices, denoted by \( S_+^d \), is clearly not a vector space since multiplying an SPD matrix by a negative scalar results in a matrix which does not belong to \( S_+^d \). Instead, \( S_+^d \) forms the interior of a convex cone in the \( d(d+1)/2 \)-dimensional Euclidean space.

The \( S_+^d \) space is most studied when endowed with a Riemannian metric and thus forms a Riemannian manifold [64]. The geodesic distance between two CovDs \( C_1, C_2 \in S_+^d \) induced by the Affine Riemannian Metric (ARM) [64] is

\(^1\)We use the publicly available code downloaded at http://www.robots.ox.ac.uk/~vgg/research/texelclass/filters.html
Dense sampling
Feature extraction from RGB cubics
Feature extraction from depth cubics
Concatenating features from RGB and depth cubics
Computing covariance matrices from all cubics

Fig. 2. The flow chart of the dense covariance descriptor extraction.

A. Dense Covariance Descriptor Extraction

Given the RGB video $C$ and depth video $D$ simultaneously recording a gesture, a set of dense 3D cubics can be sampled from them to extract local features. Let $w \times h \times t$ be the size of the sampled cubic and $l_w$, $l_h$ and $l_t$ be the sliding steps along three directions. For each pixel in the RGB cubic, we compute the following features.

1) Extracting motion features from RGB data: Let $I(x, y, t)$ be the intensity of the RGB video at pixel position $(x, y, t)$ and $u(x, y, t) = [u, v]^T$ the corresponding optical flow vector. From the intensity and optical flow vector, we compute the following feature vector for the pixel

$$F_m(x, y, t) = [u, v, u_t, v_t, Div, Vor, Gten, Sten]$$

(6)

where $u_t$ and $v_t$ are the 1-st order partial derivatives of $u$ and $v$ with respect to $t$. $Div$, $Vor$, $Gten$ and $Sten$ are the divergence, vorticity and two tensor invariants of the optical flow, respectively. In particular, $Div$ is the spatial divergence of the optical flow and can be computed as

$$Div(x, y, t) = \frac{\partial u(x, y, t)}{\partial x} + \frac{\partial v(x, y, t)}{\partial y}$$

(7)

Divergence captures the amount of local expansion in the fluid which can indicate gesture differences. $Vor$ is the vorticity of the flow field and can be computed as

$$Vor(x, y, t) = \frac{\partial v(x, y, t)}{\partial x} - \frac{\partial u(x, y, t)}{\partial y}$$

(8)
Vorticity is used to measure local spin around the axis perpendicular to the plane of the flow field, which potentially captures locally circular motions of a moving pixel. To compute $Gten$ and $Sten$, we need to introduce two matrices, namely the gradient tensor $\nabla u(x,y,t)$ and the rate of strain tensor $S(x,y,t)$

$$\nabla u(x,y,t) = \begin{pmatrix} \frac{\partial u(x,y,t)}{\partial x} \\ \frac{\partial u(x,y,t)}{\partial y} \end{pmatrix}, \quad S(x,y,t) = \frac{1}{2} (\nabla u(x,y,t) + \nabla^\top u(x,y,t))$$

$Gten$ and $Sten$ are tensor invariants that remain constant no matter what co-ordinate system they are referenced in. They are defined in terms of $\nabla u(x,y,t)$ and $S(x,y,t)$ as follows:

$$Gten(x,y,t) = \frac{1}{2} (\text{tr}^2(\nabla u(x,y,t)) - \text{tr}(\nabla^2 u(x,y,t)))$$

$$Sten(x,y,t) = \frac{1}{2} (\text{tr}^2(S(x,y,t)) - \text{tr}(S^2(x,y,t)))$$

where $\text{tr}(\cdot)$ denotes the trace operation. $Gten$ and $Sten$ are scalar properties that combine gradient tensor components thus accounting for local fluid structures.

2) Extracting textural feature from depth data: Let $D(x,y,t)$ be the intensity value of the depth video at pixel position $(x,y,t)$. To exploit the depth information, we propose to extract 8 textural features [62] for each pixel. The textural feature vector for pixel at $(x,y,t)$ is computed as follows:

$$f^i(x,y,t) = [f_1(x,y,t), f_2(x,y,t), \ldots, f_8(x,y,t)]$$

where $f_1(x,y,t), f_2(x,y,t), f_3(x,y,t)$ and $f_4(x,y,t), f_5(x,y,t), f_6(x,y,t), f_7(x,y,t)$ are the maximum responses across 6 orientations over 3 scales for two anisotropic filters, respectively. The remaining two features $f_7(x,y,t)$ and $f_8(x,y,t)$ are the responses of a Gaussian and a Laplacian of Gaussian filters both with $\sigma = 10$, respectively. Note that these 8-dimensional features can be extracted efficiently using a fast anisotropic Gaussian filter\(^2\) as shown in [63].

3) Local Covariance descriptor computation: The final feature vector of the pixel at position $(x,y,t)$ can be obtained by concatenating the motion features extracted from RGB video and the textural features extracted from the depth video as well as the position coordinates

$$f(x,y,t) = [x,y,t, f^m(x,y,t), f^f(x,y,t)]^\top \in \mathbb{R}^d$$

where $d = 19$.

For any spatio-temporal cubic, let $F = [f_1 f_2 \ldots f_m]$ be a $d \times m$ matrix, obtained by stacking $m$ feature vectors $f_i \in \mathbb{R}^d$ extracted from all pixels inside the cubic. The covariance descriptor $C$, as the name implies, is defined as

$$C = \frac{1}{m-1} \sum_{i=1}^{m} (f_i - \mu)(f_i - \mu)^\top,$$

where $\mu = \frac{1}{m} \sum_{i=1}^{m} f_i$ is the mean of the feature vectors.

\(^2\)We use the publicly available code downloaded at http://www.robots.ox.ac.uk/~vgg/research/texclass/filters.html

B. Goodbook learning on SPD Manifold

To learn a dictionary for sparse coding on SPD manifold, we operate in the kernel space via the Stein kernel with the associated mapping function $\phi$. Let $\Phi(\mathbb{X}) = [\phi(\mathbb{X}_1), \ldots, \phi(\mathbb{X}_N)]$. The dictionary learning problem can be formulated as the following minimisation problem:

$$\arg \min_{H} \|\Phi(\mathbb{X}) - \Phi(\mathbb{D})H\|_F^2 + \lambda \|H\|_1.$$  \hspace{1cm} (14)

where the $i$-th column of matrix $H \in \mathbb{R}^{K \times N}$ is the coefficients associated with $i$-th training sample $\mathbb{X}_i$, $\lambda$ is a regularization parameter. The $\ell_2$-reconstruction error measures the quality of the approximation while the complexity is measured by the $\ell_1$-norm of the optimal $H$.

To solve Eq. (14), here we assume that the dictionary atoms lie within the subspace spanned by the input data, then we can write $\Phi(\mathbb{D})$ as a linear combination of $\Phi(\mathbb{X}) = [\phi(\mathbb{X}_1), \ldots, \phi(\mathbb{X}_N)]$. Formally, this means $\Phi(\mathbb{D}) = \Phi(\mathbb{X})A$, where $A \in \mathbb{R}^{N \times K}$ is the referred to as the atom representation dictionary.

1) Sparse Coding Phase: Given training samples atom representation dictionary $A$, solving sparse coding coefficients for sample $\mathbb{X}$ can be formulated as:

$$\arg \min_h \|\phi(\mathbb{X}) - \Phi(\mathbb{X})Ah\|_2^2 + \lambda \|h\|_1.$$  \hspace{1cm} (15)

This above problem can be solved in many ways. Here, we describe an algorithm based on coordinate descent. We cyclically update over individual coordinates in $h = (h_1, \ldots, h_K)$. Define, $r = \phi(\mathbb{X}_i) - \Phi(\mathbb{X})\sum_{j=1,j \neq i}^{K} A_j h_j$. The update for $h_i$ keeping all other variables in $h$ fixed becomes:

$$\arg \min_{h_i} \|r - \Phi(\mathbb{X}) A_i h_i\|_2^2 + \lambda |h_i|$$

The above optimization problem has a closed form minimum given by

$$h_i = \text{soft}(A_i^\top \Phi(\mathbb{X})^\top r, \lambda/2)$$

where $A_i^\top \Phi(\mathbb{X})^\top r$ can be computed as $A_i^\top \phi(\mathbb{X})^\top \phi(\mathbb{X}_i) - A_i^\top \phi(\mathbb{X})^\top \phi(\mathbb{X})\sum_{j=1,j \neq i}^{K} A_j h_j$.

2) Dictionary Learning Phase: Using the above representation of $\Phi(\mathbb{D}) = \Phi(\mathbb{X})A$, Eq. (14) can be rewritten as

$$\arg \min_{A,H} \|\Phi(\mathbb{X}) - \Phi(\mathbb{X})AH\|_F^2 + \lambda \|H\|_1.$$  \hspace{1cm} (16)

Let $A_i$ denote the $i$th column in $A$. We again optimize cyclically over individual $A_i$ and $h_i$ variables while keeping all other variables fixed.

**Updating $h_i$:** Holding all variables except $h_i$ fixed, the dictionary learning problem (16) can be reduced to:

$$\arg \min_{h_i \in \mathbb{R}^n} \|P - \Phi(\mathbb{X})A_i h_i\|_F^2 + \lambda \|h_i\|_1.$$  \hspace{1cm} (17)

where $P = \Phi(\mathbb{X}) - \Phi(\mathbb{X})(\sum_{j \neq i} A_j h_j^\top)$ is the residual matrix independent of $A_i$ and $h_i$. Note that $P$ is the difference of a sparse matrix and rank one matrices. While $P$ can possibly be a dense matrix, we never need to
evaluate it explicitly. In particular, our algorithm only needs to compute matrix vector products against \( P \), namely
\[
P^\top \Phi(X)A_i = \Phi(X)^\top \Phi(X)A_i - \sum_{j \neq i} h_j A_j^\top \Phi(X)A_i.
\]
We can use matrix-vector computations to evaluate \( P^\top A_i \) without evaluating \( P \) explicitly.

To finish the update equations for \( h_i \) note that
\[
\|P - \Phi(X)A_i h_i^\top\|_F^2 = \text{Tr}((P - \Phi(X)A_i h_i^\top)(P - \Phi(X)A_i h_i^\top)^\top).
\]

Therefore, Eq. 17 can be rewritten as
\[
\arg \min_{h_i \in \mathbb{R}^n} \|P - \Phi(X)A_i h_i^\top\|_F^2 + \lambda \|h_i\|_1 \\
\equiv \arg \min_{h_i \in \mathbb{R}^n} \|P - \Phi(X)A_i h_i^\top\|_F^2 \\
\equiv \arg \min_{h_i \in \mathbb{R}^n} \text{Tr}((P - \Phi(X)A_i h_i^\top)(P - \Phi(X)A_i h_i^\top)^\top) + \lambda \|h_i\|_1,
\]
\[
= \arg \min_{h_i \in \mathbb{R}^n} \text{Tr} \left( \left( h_i - \frac{P^\top \Phi(X)A_i}{\|\Phi(X)A_i\|_2^2} \right) \left( h_i - \frac{P^\top \Phi(X)A_i}{\|\Phi(X)A_i\|_2^2} \right)^\top \right) + \lambda \|h_i\|_1 \\
= \arg \min_{h_i \in \mathbb{R}^n} \left\| h_i - \frac{P^\top \Phi(X)A_i}{\|\Phi(X)A_i\|_2^2} \right\|_1^2 + \lambda \|h_i\|_1 \\
= \arg \min_{h_i \in \mathbb{R}^n} \left\| h_i - \frac{P^\top \Phi(X)A_i}{\|\Phi(X)A_i\|_2^2} \right\|_1^2.
\]

Using Lemma 1, we get that the minimizer of Eq. 17 is
\[
h_i = \text{soft} \left( \frac{P^\top \Phi(X)A_i}{\|\Phi(X)A_i\|_2^2} \frac{\lambda}{2\|\Phi(X)A_i\|_2^2} \right).
\]

**Updating \( A_i \):** Firstly note that for updating the dictionary, Eq. (16) can be reduced to (as there is no regularization term):
\[
\arg \min_{A \in \mathbb{R}^{N \times K}} \|\Phi(X) - \Phi(X)AH\|_F^2.
\]

Holding all variables except \( A_i \) fixed, the above dictionary learning problem can be reduced to:
\[
\arg \min_{A \in \mathbb{R}^{N \times K}} \|P - \Phi(X)A_i h_i^\top\|_F^2.
\]

Like in the update of \( h_i \), after a simple numerical manipulation, Eq. 20 can be reduced as
\[
\arg \min_{A \in \mathbb{R}^{N \times K}} \left\| \Phi(X)A_i - Ph_i \right\|_1.
\]

The gradient of \( \|\Phi(X)A_i - Ph_i\|_1 \) with respect to \( A_i \) is \( 2\Phi(X)^\top (\Phi(X)A_i - Ph_i)/\|h_i\|^2 \). Given the gradient, we can use gradient descent technique to solve the above minimization problem. The idea is to iteratively set
\[
A_i^{(l)} = A_i^{(l-1)} - \alpha \left( 2\Phi(X)^\top \Phi(X)A_i^{(l-1)} - \frac{2\Phi(X)^\top Ph_i}{\|h_i\|^2} \right),
\]
where \( A_i^{(l)} \) is the value of the variable \( A_i \) in the \( l \)th iteration and \( \alpha \) is the step size. Note that using the definition of \( P \)
\[
\Phi(X)^\top Ph_i = \Phi(X)^\top \Phi(X)h_i - \Phi(X)^\top \Phi(X) \sum_{j \neq i} A_j h_j^\top h_i.
\]

**Algorithm 1: Dictionary learning on SPD manifold**

**Input:** SPD matrices \( \Phi(X) = [\phi(X_1), \ldots, \phi(X_N)] \)

**Output:** \( A \)

1. Initialize \( A_i^{(0)}, A_j^{(0)}, \ldots, A_K^{(0)} \);
2. while not(converge) do
   1. Updating \( h_i^1 \) (ith row of \( H \)) using Eq. 18;
   2. Updating \( A_i \) (ith column of \( A \)) by first solving Eq. 22 and then iteratively performing
   3. for \( l = 1 \) to \( L \) do
      1. Solving Eq. 21.
   4. end
3. end

Fig. 3. An illustration of the spatial pyramid division of the video grid.

The detailed algorithm is summarized in Algorithm 1. **Convergence of Algorithm 1:** Using a general result on convergence of Block Coordinate Descent, from [66], we can show that the limit point returned by Algorithm 1 is a stationary point of the objective function (16). In our experiments, we found that it averagely takes 10 iterations to converge.

**C. Spatial Pyramid BoW Representation and Classification**

Given a pair of RGB and depth videos from the \( k \)-th class, we adopt the spatial pyramid BoW representation to describe the global appearance of the video pair. As shown in Fig. 3, the video grid is divided into a set of cells at three scales. For example, \( 2^{3 \times 3} \) cells can be obtained at scale \( s \in \{0, 1, 2\} \). Therefore, a total of \( \sum_{s=0}^{2} 2^{3 \times 3} s \) can be obtained. Let \( C_{c,s}^{(k)} = C_{c,s,1}^{(k)} C_{c,s,2}^{(k)} \cdots \) denote the covariance descriptors computed from \( N_{c,s} \) cubes inside the \( c \)-th cell at scale \( s \) from video pair of the \( k \)-th class. Using the sparse coding method introduced in subsection IV-B, we can compute the coefficient vectors for these covariance descriptors. Similarly, let \( a_{c,s,i}^{(k)} \) denote the corresponding coefficient vectors. We can compute a histogram like representation for this cell as
\[
h_{c,s}^{(k)} = \sum_{i=1}^{N_{c,s}} a_{c,s,i}^{(k)}
\]

Therefore, the training video pair from the \( k \)-th class can be finally represented by a set of histograms \( \{h_{c,s} \mid s = 0, 1, 2, c = 1, 2, \ldots, 2^{3 \times 3} \} \).

At the recognition stage, given a test video pair, we can compute a set of histograms from this video pair using the
method above as \( \{h^s_{c,s} | s = 0, 1, 2, c = 1, 2, \ldots, 2^{3 \times s} \} \). We adopt the nearest neighbour classifier for recognition. The distance between the test sample and the training sample of the \( k \)-th class is computed as the \( \chi^2 \) distance \([67]\) between their pairwise histograms

\[
\rho(k) = \sum_{s=0}^{2^{3 \times s}} \sum_{c=0}^{B} \sum_{b=1}^{B} \frac{(h^s_{c,s}(b) - h^{(k)}_{c,s}(b))^2}{h^s_{c,s}(b) + h^{(k)}_{c,s}(b)}
\]

(24)

Then the test sample is classified as the class which has the minimal distance.

V. EXPERIMENTS

In this section, we evaluate the proposed one-shot learning gesture recognition method on the Chalearn gesture challenge dataset. We also validate the effectiveness of the proposed feature extraction methods based on bag of manifold words on a new RGB-D action recognition dataset. In the following, we present the experiment setup and results in detail.

A. Database

To validate the performance of the proposed method for one-shot learning gesture recognition, we compared our method with other state-of-the-art methods on the Chalearn gesture challenge dataset. Similar to \([23]\), we also use 20 development batches (devel01 \( \sim \) devel20), 20 validation batches (valid01 \( \sim \) valid20) and 20 final batches (final01 \( \sim \) final40) for testing. Each batch has a total of 47 gesture videos, which are split into a training set and test set. The training set includes a small set of vocabulary spanning from 8 to 15 gestures. Each test video contains 1 to 5 gestures. Detailed descriptions of the dataset can be found in \([25]\). All the samples are recorded with a Microsoft Kinect camera which provides both RGB and depth video clips.

B. Metric of Evaluation

We adopt the same metric of evaluation used in \([25]\) which uses the Levenshtein distance to calculate the score between the classified labels and the ground truth labels. The Levenshtein distance between two strings is defined as the minimum number of operations (insertions, substitutions or deletions) needed to transform one string to the other. In our evaluation, one string contains the classified labels in all samples and the other string contains their ground truth labels. For all comparisons, we compute the mean Levenshtein distance (MLD) over all video clips and batches. Note that the smaller the MLD score is, the better the performance of an algorithm is.

C. Testing with different parameters

To evaluate the performance of the proposed method using different parameters, we first keep \( \lambda = 0.02 \) and change the values of \( B \) from 500 to 3500. Fig. 4 shows the performance when different \( B \) are used, from which we can see that the value of the parameter \( B \) significantly affects the performance when \( B \) increases from a smaller value 500 to a relatively larger value 3000 and then tends to be stable after being 3000. As well known the parameter \( \lambda \) in the sparse coding step controls how important the sparsity constraint is relative to the reconstruction error. As shown in Fig. 5, the MLD score has a significant drop when \( \lambda \) changes from 0.0001 to 0.005 and then achieves stable with little disturbance when \( \lambda \) changing from 0.005 to 0.03.

The other factor that may affect the performance is the pyramid structure. To test the performance of the proposed method when different pyramid divisions are used, we compute the MLD scores with parameters \( B = 3000 \) and \( \lambda = 0.02 \) but when different combinations of pyramid scales, e.g., \( \{0\}, \{1\}, \{2\}, \{0, 1\}, \{0, 2\}, \{1, 2\}, \{0, 1, 2\} \) are used. The obtained MLD scores are shown in Table 1. From this table, we can see that when the BoW representation is computed from the whole video grid, the performance is very poor. If dividing the video grid into a set of cells and then use the BoW histograms computed from all cells for the classification, the performance is significantly increased. We can also see that all three scales play important roles in improving the final performance.

D. Testing the effectiveness of key components

The proposed method (BoMW) has two key components including 1) extracting covariance descriptor to encode both the RGB and depth information and 2) learning codebook on SPD manifold. To validate the effectiveness of each key
component, we conducted several experiments to test the performance when a key component is replaced by other method. For example, to validate the effectiveness of the covariance descriptor extracted on both the RGB and depth information, we compare the performance of the proposed method when the covariance descriptor is extracted only from the RGB information or depth information. To validate the effectiveness of the proposed codebook learning method, we compare the proposed method with a SPD codebook learning method which uses the Log-Euclidean operator [68] to project covariance matrices to Euclidean vector space and then uses standard vector learning methods to learn the codebook. We use the abbreviation BoMW-RGB and BoMW-D to denote the proposed method using RGB and depth information, respectively. The abbreviation BoMW-LogE denotes the proposed method using Log-Euclidean operator to project covariance matrices to Euclidean vector space. Table II presents the experiment results of testing the effectiveness of key components. As we can see from Table II, when RGB or depth information is used alone, the resulting performance is worse than the proposed method when both RGB and depth information are used. By comparing RGB and depth information, we find that RGB information is important than depth information especially when there is significant contrast between the foreground and background. An example gesture is shown in Fig. 6(a). However, we also find that in some cases RGB information is not capable of discriminating different gestures. For example, in Fig. 6(b), the fingers of the subject have very similar colour information with the background and therefore when only RGB information is used, the proposed method fails to recognise the gesture. When the Log-Euclidean operator is used to project covariance matrices to Euclidean vector space, the resulting performance is worse than the proposed method using the Stein kernel to project covariance matrices to the RKHS.

**TABLE II**

<table>
<thead>
<tr>
<th>Method</th>
<th>MLD</th>
</tr>
</thead>
<tbody>
<tr>
<td>BoMW-RGB</td>
<td>0.1725</td>
</tr>
<tr>
<td>BoMW-D</td>
<td>0.1958</td>
</tr>
<tr>
<td>BoMW-LogE</td>
<td>0.1538</td>
</tr>
<tr>
<td>BoMW</td>
<td>0.1396</td>
</tr>
</tbody>
</table>

**E. Comparison with recent representative methods**

We compare the proposed BoMW with several state-of-the-art feature extraction methods for gesture extraction including MoSIFT [39], 3D MoSIFT [38], 3D EMoSIFT [23] and Multi-Layered Features (MLF) [24]. All the results of the compared methods were obtained by running their methods on the same batches with the same training/testing splits. The compared results are shown in Table III, from which we can see that the proposed method is significantly superior to the MoSIFT method. It should be noted that MoSIFT was not originally proposed for gesture recognition from RGB-D data. In our experiments, we just use MoSIFT to extract features from both RGB and depth videos, respectively and then concatenate them together. Compared to three methods specifically designed for RGB-D data, the proposed method only has slight superiority in term of MLD. But we have to emphasize that the proposed method does not depend on the accurate segmentation of user’s body and fingers, which makes our methods suitable for more complicated scenarios. In Tables IV and V, we also show the MLD scores of the compared methods on each development batch as well as each final batch. As shown in Table IV, in batches 2, 18 and 20, the proposed method does not achieve the best performance. However, the proposed method achieves the best performance in the remaining 17 batches. As shown in Table IV, in batches 24, 27, 30 and 34, the proposed method does not achieve the best performance. However, the proposed method achieves the best performance in the remaining 16 batches.

**F. Evaluating on action recognition**

In this section, we evaluate the effectiveness of the proposed feature extraction method for RGB-D action recognition. There are several large RGB-D action recognition datasets, such as MSR gesture 3D [69], NTU RGB+D [26], MSRC-12 [70], which can also be used to evaluated whether the
TABLE III
EXPERIMENT RESULTS OF THE COMPARED METHODS.

<table>
<thead>
<tr>
<th>Method</th>
<th>MLD</th>
</tr>
</thead>
<tbody>
<tr>
<td>MoSIFT [39]</td>
<td>0.4572</td>
</tr>
<tr>
<td>3D MoSIFT [38]</td>
<td>0.1837</td>
</tr>
<tr>
<td>3D EmoSIFT [23]</td>
<td>0.1629</td>
</tr>
<tr>
<td>MLF [24]</td>
<td>0.1483</td>
</tr>
<tr>
<td>BoMW</td>
<td>0.1396</td>
</tr>
</tbody>
</table>

TABLE IV
MLD SCORES OF THE COMPARED METHODS ON EACH BATCH (VALID01-VALID20).

<table>
<thead>
<tr>
<th>Batch</th>
<th>MoSIFT</th>
<th>3D MoSIFT</th>
<th>3D EmoSIFT</th>
<th>MLF</th>
<th>BoMW</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.4012</td>
<td>0.1723</td>
<td>0.1421</td>
<td>0.1297</td>
<td>0.1201</td>
</tr>
<tr>
<td>2</td>
<td>0.3825</td>
<td>0.1539</td>
<td>0.1287</td>
<td>0.1102</td>
<td>0.1192</td>
</tr>
<tr>
<td>3</td>
<td>0.4831</td>
<td>0.2013</td>
<td>0.1823</td>
<td>0.1592</td>
<td>0.1472</td>
</tr>
<tr>
<td>4</td>
<td>0.4012</td>
<td>0.1796</td>
<td>0.1538</td>
<td>0.1321</td>
<td>0.1290</td>
</tr>
<tr>
<td>5</td>
<td>0.3927</td>
<td>0.1539</td>
<td>0.1291</td>
<td>0.1023</td>
<td>0.1021</td>
</tr>
<tr>
<td>6</td>
<td>0.4896</td>
<td>0.1876</td>
<td>0.1673</td>
<td>0.1490</td>
<td>0.1401</td>
</tr>
<tr>
<td>7</td>
<td>0.3536</td>
<td>0.1384</td>
<td>0.1462</td>
<td>0.1261</td>
<td>0.1224</td>
</tr>
<tr>
<td>8</td>
<td>0.4972</td>
<td>0.2017</td>
<td>0.2139</td>
<td>0.1823</td>
<td>0.1629</td>
</tr>
<tr>
<td>9</td>
<td>0.4793</td>
<td>0.1945</td>
<td>0.1733</td>
<td>0.1521</td>
<td>0.1491</td>
</tr>
<tr>
<td>10</td>
<td>0.4586</td>
<td>0.1838</td>
<td>0.1654</td>
<td>0.1467</td>
<td>0.1402</td>
</tr>
<tr>
<td>11</td>
<td>0.5965</td>
<td>0.1487</td>
<td>0.1374</td>
<td>0.1231</td>
<td>0.1031</td>
</tr>
<tr>
<td>12</td>
<td>0.3912</td>
<td>0.1526</td>
<td>0.1412</td>
<td>0.1245</td>
<td>0.1125</td>
</tr>
<tr>
<td>13</td>
<td>0.3627</td>
<td>0.1487</td>
<td>0.1401</td>
<td>0.1294</td>
<td>0.1099</td>
</tr>
<tr>
<td>14</td>
<td>0.4625</td>
<td>0.1987</td>
<td>0.1721</td>
<td>0.1581</td>
<td>0.1461</td>
</tr>
<tr>
<td>15</td>
<td>0.4135</td>
<td>0.1625</td>
<td>0.1524</td>
<td>0.1321</td>
<td>0.1293</td>
</tr>
<tr>
<td>16</td>
<td>0.4427</td>
<td>0.1793</td>
<td>0.1613</td>
<td>0.1481</td>
<td>0.1352</td>
</tr>
<tr>
<td>17</td>
<td>0.4824</td>
<td>0.2053</td>
<td>0.1856</td>
<td>0.1780</td>
<td>0.1762</td>
</tr>
<tr>
<td>18</td>
<td>0.5026</td>
<td>0.2169</td>
<td>0.1974</td>
<td>0.1824</td>
<td>0.1831</td>
</tr>
<tr>
<td>19</td>
<td>0.4987</td>
<td>0.2178</td>
<td>0.2012</td>
<td>0.2013</td>
<td>0.1925</td>
</tr>
<tr>
<td>20</td>
<td>0.4724</td>
<td>0.1926</td>
<td>0.1798</td>
<td>0.1723</td>
<td>0.1837</td>
</tr>
</tbody>
</table>

TABLE V
MLD SCORES OF THE COMPARED METHODS ON EACH BATCH (FINAL21-FINAL40).

<table>
<thead>
<tr>
<th>Batch</th>
<th>MoSIFT</th>
<th>3D MoSIFT</th>
<th>3D EmoSIFT</th>
<th>MLF</th>
<th>BoMW</th>
</tr>
</thead>
<tbody>
<tr>
<td>21</td>
<td>0.3942</td>
<td>0.1699</td>
<td>0.1499</td>
<td>0.1333</td>
<td>0.1298</td>
</tr>
<tr>
<td>22</td>
<td>0.3714</td>
<td>0.1603</td>
<td>0.1187</td>
<td>0.1203</td>
<td>0.1132</td>
</tr>
<tr>
<td>23</td>
<td>0.4626</td>
<td>0.2132</td>
<td>0.1749</td>
<td>0.1361</td>
<td>0.1391</td>
</tr>
<tr>
<td>24</td>
<td>0.4132</td>
<td>0.1732</td>
<td>0.1584</td>
<td>0.1421</td>
<td>0.1450</td>
</tr>
<tr>
<td>25</td>
<td>0.3893</td>
<td>0.1603</td>
<td>0.1301</td>
<td>0.1211</td>
<td>0.1121</td>
</tr>
<tr>
<td>26</td>
<td>0.4625</td>
<td>0.1814</td>
<td>0.1598</td>
<td>0.1356</td>
<td>0.1309</td>
</tr>
<tr>
<td>27</td>
<td>0.3618</td>
<td>0.1226</td>
<td>0.1502</td>
<td>0.1198</td>
<td>0.1213</td>
</tr>
<tr>
<td>28</td>
<td>0.4825</td>
<td>0.1941</td>
<td>0.2088</td>
<td>0.1725</td>
<td>0.1569</td>
</tr>
<tr>
<td>29</td>
<td>0.4613</td>
<td>0.1899</td>
<td>0.1793</td>
<td>0.1613</td>
<td>0.1511</td>
</tr>
<tr>
<td>30</td>
<td>0.4604</td>
<td>0.1802</td>
<td>0.1705</td>
<td>0.1527</td>
<td>0.1532</td>
</tr>
<tr>
<td>31</td>
<td>0.4012</td>
<td>0.1511</td>
<td>0.1401</td>
<td>0.1331</td>
<td>0.1124</td>
</tr>
<tr>
<td>32</td>
<td>0.3817</td>
<td>0.1490</td>
<td>0.1523</td>
<td>0.1300</td>
<td>0.1209</td>
</tr>
<tr>
<td>33</td>
<td>0.3725</td>
<td>0.1414</td>
<td>0.1519</td>
<td>0.1312</td>
<td>0.1115</td>
</tr>
<tr>
<td>34</td>
<td>0.4705</td>
<td>0.1904</td>
<td>0.1844</td>
<td>0.1609</td>
<td>0.1629</td>
</tr>
<tr>
<td>35</td>
<td>0.4213</td>
<td>0.1705</td>
<td>0.1624</td>
<td>0.1415</td>
<td>0.1390</td>
</tr>
<tr>
<td>36</td>
<td>0.4329</td>
<td>0.1811</td>
<td>0.1636</td>
<td>0.1516</td>
<td>0.1425</td>
</tr>
<tr>
<td>37</td>
<td>0.4793</td>
<td>0.1997</td>
<td>0.1725</td>
<td>0.1890</td>
<td>0.1839</td>
</tr>
<tr>
<td>38</td>
<td>0.4914</td>
<td>0.2104</td>
<td>0.1800</td>
<td>0.1876</td>
<td>0.1725</td>
</tr>
<tr>
<td>39</td>
<td>0.4895</td>
<td>0.2113</td>
<td>0.1923</td>
<td>0.2123</td>
<td>0.1834</td>
</tr>
<tr>
<td>40</td>
<td>0.4810</td>
<td>0.2184</td>
<td>0.1815</td>
<td>0.1800</td>
<td>0.1706</td>
</tr>
</tbody>
</table>

TABLE VI
EXPERIMENT RESULTS OF THE TWO EVALUATION SETTINGS ON THE NTU RGB+D DATASET.

<table>
<thead>
<tr>
<th>Method</th>
<th>Cross Subject Accuracy</th>
<th>Cross View Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>HOG</td>
<td>32.24%</td>
<td>22.27%</td>
</tr>
<tr>
<td>Super Normal Vector</td>
<td>31.82%</td>
<td>13.61%</td>
</tr>
<tr>
<td>HON4D</td>
<td>30.56%</td>
<td>7.26%</td>
</tr>
<tr>
<td>P-LSTM</td>
<td>62.93%</td>
<td>48.25%</td>
</tr>
<tr>
<td>BoMW</td>
<td>35.25%</td>
<td>36.18%</td>
</tr>
</tbody>
</table>

VI. CONCLUSION

In this paper, we propose a novel feature extraction method for gesture recognition, namely, bag of manifold words (BoMW), which uses covariance matrices to combine both RGB and depth features from local spatio-temporal blocks. Since covariance matrices are SPD matrices, which spans a SPD manifold. We further propose a novel sparse coding method on SPD manifolds and encode covariance matrices as the final feature representation in a bag of word fashion. The nearest neighbour classifier is finally adopted to perform the one-shot learning gesture recognition. Experimental results
on the ChaLearn gesture dataset demonstrate the outstanding performance compared to several state-of-the-art methods. The effectiveness of the proposed feature extraction method is also validated on a new RGB-D action recognition dataset.

REFERENCES


Feng Jiang received the B.S., M.S., and Ph.D. degrees in computer science from Harbin Institute of Technology (HIT), Harbin, China, in 2001, 2003, and 2008, respectively. He is now an Associated Professor in the Department of Computer Science, HIT. He had been a visiting scholar in the School of Electrical Engineering, Princeton University, United States. His research interests include computer vision, image and video processing and pattern recognition.

Yuankai Qi received the B.S. and M.S. degrees from Harbin Institute of Technology, China, in 2011 and 2013, respectively. He is currently working toward his Ph.D. degree in the School of computer science and technology, Harbin Institute of Technology, China. His research interests include object tracking, sparse coding, and machine learning.

Jun Zhang received her B.S., M.S., and Ph.D degrees in School of Computer Science and Information Engineering from Hefei University of Technology (HFUT), China in 2007, 2009 and 2013, respectively. From August 2010 to September 2012, and May 2015 to September 2015, she worked at the Department of Cognitive, Linguistic & Psychological Sciences (CLPS) at Brown University as a visiting research fellow. From July 2013 until now, she is a postdoctoral fellow in computer science, and also acts as a associate professor in School of Computer Science and Information Engineering of HFUT. Her current interests include computer vision, vision perception, and pattern recognition.

Yuliang Guo received his B.S.M.S degree in material science from Shanghai Jiaotong University, Shanghai, China, in 2009, M.S.C.E in computer engineering from Brown University, Rhode Island, USA in 2011, and is currently a Ph.D candidate in computer engineering at Brown University. His research focus on computer vision, specifically in exploiting graphical models integrated with geometric constraints, and applying to edge detection, contour extraction, animal pose detection and behavior analysis.

Huiyu Zhou obtained a Bachelor of Engineering degree in Radio Technology from the Huazhong University of Science and Technology of China, and a Master of Science degree in Biomedical Engineering from the University of Dundee of United Kingdom, respectively. He was then awarded a Doctor of Philosophy degree in Computer Vision from the Heriot-Watt University, Edinburgh, United Kingdom. Dr. Zhou is an assistant professor at School of Electronics, Electrical Engineering and Computer Science, Queens University of Belfast, United Kingdom. He has published over 130 peer-reviewed papers in the field. His research work has been or is being supported by UK EPSRC, EU, Royal Society, Leverhulme Trust, Puffin Trust, Invest NI and industry.